

ABSTRACT

Title of Document: TRANSPORT PROPERTIES AND MELT
DISTRIBUTION OF PARTIALLY MOLTEN
MANTLE ROCKS: INSIGHTS FROM MICRO-
COMPUTED TOMOGRAPHY AND
VIRTUAL ROCK PHYSICS SIMULATIONS

Kevin John Miller, Doctor of Philosophy, 2015

Directed By: Associate Professor Wen-lu Zhu
Associate Professor Laurent G.J. Montési
Department of Geology

Mid-ocean ridges are a fundamental component of plate tectonics on Earth. They are the longest mountain ranges; combined, they stretch over 70,000 km of the Earth's surface. They are significant sources of volcanism, producing more than 20 km³ of new oceanic crust each year. The volcanism observed at the ridge axis is linked to processes that transport and focus melt in the underlying upper mantle.

Typically, upper mantle melt distribution is inferred either through inversion of geophysical data, such as electromagnetic signals, or through geodynamic modeling. Both approaches require robust constitutive relationship between on electrical conductivity, permeability, and porosity. Unfortunately, direct measurements of transport properties of partially molten rock are technically

challenging due to the extreme conditions required for melting. This work aims to quantify permeability-porosity and electrical conductivity-porosity relationships of partially molten monomineralic and polymineralic aggregates by simulating fluid flow and direct current within experimentally obtained, high-resolution, three-dimensional (3-D) microstructures of partially molten rocks.

In this study, I synthesized rocks containing various proportions of olivine, orthopyroxene (opx), and basaltic melt, common components of the upper mantle. I imaged their 3-D microstructure using high-resolution, synchrotron-based X-ray micro-computed tomography. The resulting 3-D geometries constitute virtual rock samples on which pore morphology, permeability, and electrical conductivity were numerically quantified.

This work yields microstructure-based electrical conductivity-porosity and permeability-porosity power laws for olivine-melt and olivine-opx-melt aggregates containing melt fractions of 0.02 to 0.20. By directly comparing the velocity and electrical fields, which are outputs of the fluid flow and direct current simulations, respectively, this study provides strong evidence that fluid and electricity travel through distinctly different pathways within the same rock, due to the stronger dependence of fluid flux on hydraulic radius. This study also provides the first quantitative evidence of lithological melt partitioning, where melt fractions spatially associated with olivine are systematically higher than those with orthopyroxene due to the relatively low surface energy density of olivine-melt interfaces with respect to opx-melt interfaces. The results of this study place important, novel constraints on 3-

D melt distribution and transport properties of the partially molten mantle regions beneath mid-ocean ridges.

TRANSPORT PROPERTIES AND MELT DISTRIBUTION OF PARTIALLY
MOLTEN MANTLE ROCKS: INSIGHTS FROM MICRO-COMPUTED
TOMOGRAPHY AND VIRTUAL ROCK PHYSICS SIMULATIONS

By

Kevin John Miller

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2015

Advisory Committee:

Associate Professor Wen-lu Zhu, Chair (University of Maryland)
Associate Professor Laurent G.J. Montési, Co-adviser (University of Maryland)
Assistant Professor Vedran Lekic (University of Maryland)
Assistant Professor Nicholas Schmerr (University of Maryland)
Senior Research Scientist Deborah Smith (Woods Hole Oceanographic Institution)
Professor Bryan Eichhorn, Dean's Representative (University of Maryland)

© Copyright by
Kevin John Miller
2015

Preface

In this document, the transport properties, e.g. permeability and electrical conductivity, of monomineralic and polymineralic partially molten rocks are quantified and linked to volume proportion of melt. Since transport properties of partially molten rocks are notoriously difficult to measure due to the extreme pressure-temperature conditions required for melting and high viscosity of basaltic melt, I take a novel approach: imaging and digitizing synthetic partially molten mantle rocks using advanced. These 3-D images constitute virtual rock samples on which digital rock physics (DRP) experiments can be conducted and transport properties can be quantified. My methodology, results, and implications for understanding melt transport at mid-ocean ridges are discussed over the next six chapters.

- Chapter 1: I introduce geological context and outstanding questions related to melt transport at mid-ocean ridges. The concepts necessary to understand my research methods and findings are introduced.
- Chapter 2: I quantify the 3-D melt distribution and permeability of partially molten olivine-basaltic melt as a function of melt fraction.
- Chapter 3: I derive the electrical conductivity of partial melts from microstructural considerations. I compare my results with previously conducted experiments in literature.
- Chapter 4: I investigate the role of mineral heterogeneity and surface energy on melt distribution in samples containing olivine,

orthopyroxene, and basaltic melt.

- Chapter 5: I compute the permeability and electrical conductivity of partially molten rock samples composed of olivine, orthopyroxene, and basaltic melt.
- Chapter 6: Concluding remarks and a preview of future research.

Acknowledgements

This dissertation would not have been possible without the support from so many of my family, friends, and colleagues.

I want to thank my parents, brothers, and sister for their unconditional love, support, and guidance.

To my undergraduate and graduate school buddies, I want to thank you for your friendship and support. You have made my time at the University of Maryland the best years of my life. Specifically, I want to thank my friend Aram Vartanyan, a recent physics Ph.D. graduate, for all your help with learning advanced physics and numerical modeling concepts, Eric Rosenthal for your friendship and insightful discussions, and Brian Fields for your friendship and all-night study binges, from which I learned so much. I also want to thank my friends and colleagues: Harry Lisabeth for your insightful discussions about rock mechanics concepts, Mark Larson and Hailong Bai for your expertise on mid-ocean ridge processes, Jesse Wimert and Justin DeSha-Overcash for your friendship, and Tyler Drombosky who helped me to understand and internalize the applied math concepts that are included in this research.

A very special thanks goes out to my friends Ian Winston, Matt Mehlbaum, Robbie Dinneen, and D.J. Bowes. You are my family, and I will always love you guys.

Another special thanks goes out to my co-advisers Wenlu Zhu and Laurent Montési, who introduced me to some of the coolest scientific projects that I could have hoped to work on. Your patience, guidance, excellent mentorship skills, and

expert opinions have had an immensely positive impact on me. You have taught me to approach problems with a critical and creative mentality.

I want to thank the researchers at Woods Hole Oceanographic Institution, specifically my collaborator Glenn Gaetani who mentored me in the experimental petrology aspects of my research. Also, I want to thank Véronique Le Roux for your insight, expertise, and generosity with your time and scientific resources. I want to thank Emily Sarafian for your help troubleshooting the equipment in the experimental petrology lab and your expertise related to electrical conductivity of partially molten mantle rock.

I want to thank the researchers at the Advanced Photon Source, specifically Xianghui Xiao, for your insightful discussions and for giving me the opportunity to use their beamline, the coolest piece of scientific equipment I have ever worked with. Also, thank you for introducing me to the research field of image processing.

Thank you to Phil Piccoli who conducted the quantitative chemistry analysis of our harzburgite samples.

Thank you to NSF and the Ann Wylie Dissertation Fellowship for funding my research.

Thank you to all of those that I have left out. There are so many people I have met over the past five years that have made an impact on my science, personality, and way of thinking.

Table of Contents

Preface.....	ii
Acknowledgements.....	iv
Table of Contents.....	vi
List of Tables.....	ix
List of Figures.....	x
Table of Abbreviations.....	xii
 Chapter 1: Introduction.....	 1
1.1 Melting at mid-ocean ridges.....	1
1.2 Melt microstructure.....	5
1.3 Permeability of the melt microstructure.....	7
1.3.1 Permeability of idealized geometries.....	7
1.3.2 Experimental constraints on permeability.....	8
1.3.3 Electrical conductivity of partially molten mantle rocks.....	12
1.3.4 Using electrical conductivity to infer 3-D melt distribution.....	15
1.3.5 3-D melt distribution from X-ray micro-computed tomography.....	15
1.4 Basics of X-ray micro-computed tomography.....	16
1.5 Segmentation of grayscale image.....	19
1.6 Digital rock physics applied to partially molten rock: an overview.....	20
 Chapter 2: Experimental quantification of permeability of partially molten mantle rocks.....	 21
2.1 Introduction.....	22
2.2 Experimental Methods.....	25
2.2.1 Sample preparation.....	25
2.3 Analytical Methods.....	29
2.3.1 Synchrotron X-ray micro-computed tomography.....	29
2.3.2 Subvolume selection.....	31
2.3.3 Noise reduction and segmentation techniques.....	31
2.3.4 Quantification of network topology.....	33
2.3.5 Quantification of permeability.....	36
2.4 Results.....	39
2.4.1 Grain size results.....	39
2.4.2 Connectivity of the melt network.....	40
2.4.3 Permeability results.....	43
2.4.4 Permeability anisotropy.....	45
2.5 Discussion.....	47
2.5.1 Morphology of melt microstructure.....	47
2.5.2 Interpretation of power law exponent.....	47
2.5.3 1-D mantle model.....	49
2.5.4 Implications for mantle heterogeneities.....	51
2.6 Conclusion.....	53
 Chapter 3: Influence of microstructure on electrical conductivity of partially molten rocks.....	 55

3.1 Introduction.....	56
3.2 Methods.....	61
3.2.1 Sample preparation and imaging.....	61
3.2.2 Subvolume selection.....	62
3.2.3 Noise-removal and segmentation.....	62
3.2.4 Direct current simulations.....	64
3.2.5 Fluid flow simulations.....	68
3.2.6 Computing tortuosity.....	69
3.3 Results.....	69
3.3.1 Electrical conductivity.....	69
3.3.2 Permeability.....	72
3.3.3 Tortuosity.....	75
3.4 Discussion.....	75
3.4.1 Electrical conductivity and permeability comparison.....	75
3.4.2 Comparison with experimental data.....	76
3.4.3 Melt films.....	79
3.4.4 H ₂ O in melt.....	81
3.4.5 H ₂ O in olivine.....	82
3.4.6 Chemical heterogeneity.....	84
3.5 Conclusion.....	85
Chapter 4: Experimental evidence for lithologic melt partitioning between olivine and orthopyroxene in partially molten harzburgite.....	87
4.1 Introduction.....	88
4.2 Methods.....	93
4.2.1 Sample preparation of harzburgite samples.....	93
4.2.2 Imaging Procedure.....	96
4.2.3 Subvolume selection.....	98
4.2.4 Image segmentation.....	101
4.2.5 Quantification of local melt fraction distribution.....	102
4.2.6 Characterizing grain size distributions.....	105
4.3 Results.....	106
4.3.1 Visual inspection of melt distribution.....	106
4.3.2 Local melt fraction distributions.....	108
4.3.3 Grain size distribution.....	111
4.4 Discussion.....	111
4.4.1 Melt concentration due to lithologic melt partitioning.....	111
4.4.2 Lithologic melt partitioning and transport properties.....	116
4.4.3 Geological implications for lithologic melt partitioning.....	118
4.4.4 Grain size and melt fraction.....	119
4.5 Conclusion.....	120
Chapter 5: Permeability and electrical conductivity of partially molten harzburgite.....	122
5.1 Introduction.....	123
5.2 Methods.....	127

5.2.1	Sample preparation of harzburgite samples.....	127
5.2.2	Imaging procedure.....	127
5.2.3	Subvolume selection.....	128
5.2.4	Image segmentation.....	128
5.2.5	Computation of permeability and electrical conductivity.....	130
5.2.6	Characterizing grain size distributions.....	131
5.3	Results.....	132
5.3.1	Statement about uncertainty.....	132
5.3.1	Permeability.....	134
5.3.2	Electrical conductivity.....	136
5.4	Discussion.....	138
5.4.1	Influence of opx on permeability.....	138
5.4.2	Implications for trace element partitioning in xenoliths.....	140
5.4.3	Comparison between permeability and electrical conductivity.....	141
5.5	Conclusion.....	141
Chapter 6: Summary and future work.....		143
6.1	Summary of results and conclusions.....	143
6.2	Future research directions.....	145
Appendix A: Supplementary information for microstructure and permeability quantification.....		147
A.1	Removing noise and anisotropic diffusion filtering.....	147
A.2	Segmenting using watershed transformation.....	149
A.3	Determining the size of the representative volume element.....	151
A.4	Cleaning the skeletonized melt network.....	153
A.5	Time series experiment.....	154
A.6	Correcting for skeletonization artifacts.....	158
	Source code for SkeletonCleaner.m.....	160
Appendix B: Supplementary information for electrical conductivity quantification.....		191
B.1	Benchmark for bulk conductivity computation.....	191
	Source code for FDECC.....	192
Appendix C: Summary of experimental charges and methods for measuring local melt fraction distribution.....		217
C.1	Summary of harzburgite samples.....	217
C.2	Quantitative chemistry analysis for harzburgite samples.....	219
	Source Code for LPAnalyze.m.....	223
References.....		236

List of Tables

Table 2.1: Results summary for microstructural characterization and fluid flow simulations conducted on dunite suite.....	28
Table 3.1: Results summary for direct current simulations conducted on olivine-basaltic melt suite.....	70
Table 4.1: Recipe for harzburgite oxide mix.....	95
Table 4.2: Results summary for grain size and lithologic melt partitioning.....	115
Table 5.1: Results summary for permeability and electrical conductivity of harzburgite suite.....	133
Table A.1 Summary of permeability results from olivine-basaltic melt suite.....	157
Table A.2 Explanation of ScobaCleaner subroutines.....	160
Table C.1 Summary analysis conducted for olivine-opx- basaltic melt samples.....	218
Table C.2 Results of quantitative chemistry conducted on olivine-opx-basaltic melt samples.....	220

List of Figures

Figure 1.1	Schematic diagram of mid-ocean ridge and melting conditions.....	2
Figure 1.2	Schematic diagram of idealized partially molten grain pack.....	4
Figure 1.3	Coordination number histogram for an actual, 3-D melt network.....	6
Figure 1.4	Analogue experiments permeability results.....	9
Figure 1.5	Permeability experiment on olivine-basaltic melt aggregate.....	11
Figure 1.6	Interpretations of electrical conductivity mixing models.....	14
Figure 1.7	Computed tomography part 1: Radiograph.....	18
Figure 1.8	Computed tomography part 2: Filtered-back-projection.....	18
Figure 2.1	Sample preparation of olivine-basaltic melt samples.....	27
Figure 2.2	Imaging setup for olivine-basaltic melt samples.....	30
Figure 2.3	Determination of representative subvolume for fluid low simulations.....	32
Figure 2.4	Volume rendering of melt label images.....	35
Figure 2.5	Schematic diagram of fluid flow simulation.....	38
Figure 2.6	Olivine-basaltic melt equivalent diameter distributions.....	42
Figure 2.7	Olivine-basaltic melt coordination number distributions.....	44
Figure 2.8	Permeability results for olivine-basaltic melt suite.....	46
Figure 3.1	Schematic diagram of mid-ocean ridge system and porous flow.....	57
Figure 3.2	Determination of representative subvolume for direct current simulations.....	63
Figure 3.3	Schematic diagram of direct current simulations.....	66
Figure 3.4	Comparison of permeability and electrical conductivity of olivine-basaltic melt suite.....	71
Figure 3.5	Visual comparison of velocity and current density fields.....	73
Figure 3.6	Comparison of fluid and electric tortuosity of olivine- basaltic melt suite.....	74
Figure 3.7	Comparison of computed electrical conductivity and experiments, evaluation of water on bulk electrical conductivity, and influence of melt films on electrical conductivity.....	77
Figure 4.1	Explanation of lithological melt partitioning and minimum-energy melt fraction.....	90
Figure 4.2	Schematic diagram of previous lithologic fluid partitioning experiments.....	92
Figure 4.3	Schematic diagram of harzburgite experiment charges	94
Figure 4.4	Digital rock physics workflow for olivine-opx-basaltic melt suite.....	97
Figure 4.5	Visualization of olivine-opx-basaltic melt μ -CT reconstructions	99
Figure 4.6	Visualization of vertical melt fraction heterogeneity in olivine-opx-basaltic melt sample.....	100
Figure 4.7	Visualization of 3-D olivine, opx, and basaltic melt	

	label images.....	103
Figure 4.8	Schematic diagram of local melt fraction quantification technique.....	104
Figure 4.9	Visual confirmation of lithologic melt partitioning in olivine-opx-basaltic melt sample.....	107
Figure 4.10	Quantification of lithologic melt partitioning.....	109
Figure 4.11	Equivalent diameter distributions of olivine-opx-basaltic melt subvolumes.....	112
Figure 4.12	Variation of grain size linked to melt fraction.....	117
Figure 5.1	Schematic diagram of olivine and opx triple junctions.....	124
Figure 5.2	Reconstructed tomography slice from harzburgite sample.....	129
Figure 5.2	Comparison of olivine-basaltic melt and olivine-opx-basaltic melt harzburgite permeability.....	135
Figure 5.3	Comparison of olivine-basaltic melt and olivine-opx-basaltic melt harzburgite electrical conductivity.....	137
Figure 5.4	Comparison of olivine-basaltic melt and olivine-opx-basaltic melt fluid and electric tortuosity.....	139
Figure A.1	Demonstration of anisotropic diffusion filter to remove noise and image artifacts.....	148
Figure A.2	Schematic diagram demonstrating the use of the watershed transform to segment the grayscale image.....	150
Figure A.3	Selecting a suitable subvolume size based on permeability computations.....	152
Figure A.4	Time series equivalent diameter distribution for olivine-basaltic melt suite.....	155
Figure A.5	Time series coordination number distribution for olivine-basaltic melt suite.....	156
Figure C.1	Visualization of bright dendritic phase.....	222

Table of Abbreviations

Acronym or abbreviation	Full spelling
MOR	mid-ocean ridge
MORB	mid-ocean ridge basalt
μ -CT	micro-computed tomography
3-D	three-dimensional
opx	orthopyroxene
cpx	clinopyroxene
DRP	digital rock physics
MORB	mid-ocean ridge basalt
APES	Absolute Permeability Experiment Simulation
APTC	Absolute Permeability Tensor Calculation
EDD	equivalent diameter distribution
CND	coordination number distribution
SEM	scanning electron microscope
MT	magnetotelluric
EDS	electron dispersive spectroscopy

Chapter 1: Introduction

1.1 Melting at mid-ocean ridges

Melting of mantle rock is controlled by environmental conditions, such as pressure, temperature, and volatile content. For an intraplate region of the upper mantle far from sources of volcanism, the pressure-temperature conditions are generally thought to be insufficient to cause melting of the mantle, which is composed primarily of olivine and pyroxene. At mid-ocean ridges, however, divergence of the overriding oceanic plates induces a pressure gradient that pulls upward hot rock sourced deeper in the mantle. The resultant pressure drop, which occurs faster than thermal equilibration, carries the peridotite across its solidus (Fig. 1.1), inducing partial melting – also known as decompression melting – over a broad region (Allégre et al., 1973; McKenzie and Bickle, 1988). The pressure and temperature conditions in the upper mantle, which vary with depth, define a prism-shaped region of partial melt more or less centered about the ridge axis that extends laterally for hundreds of kilometers (McKenzie and Bickle, 1988; Oxburgh, 1980). Seismic (e.g. MELT Seismic Team, 1998; Toomey et al., 1998). Magnetotelluric (e.g. Evans et al., 1999; Key et al., 2013) surveys of the fast-spreading East Pacific Rise confirmed this conceptual model of mid-ocean ridge melting. However, the mechanism for transporting and focusing melt to the ridge axis is still debated.

A number of theories have been proposed to explain the transport of melt from depth to the ridge, which involves both the ascent of melt and redirection to the ridge axis (melt focusing). For a long time, it was thought that the same pressure

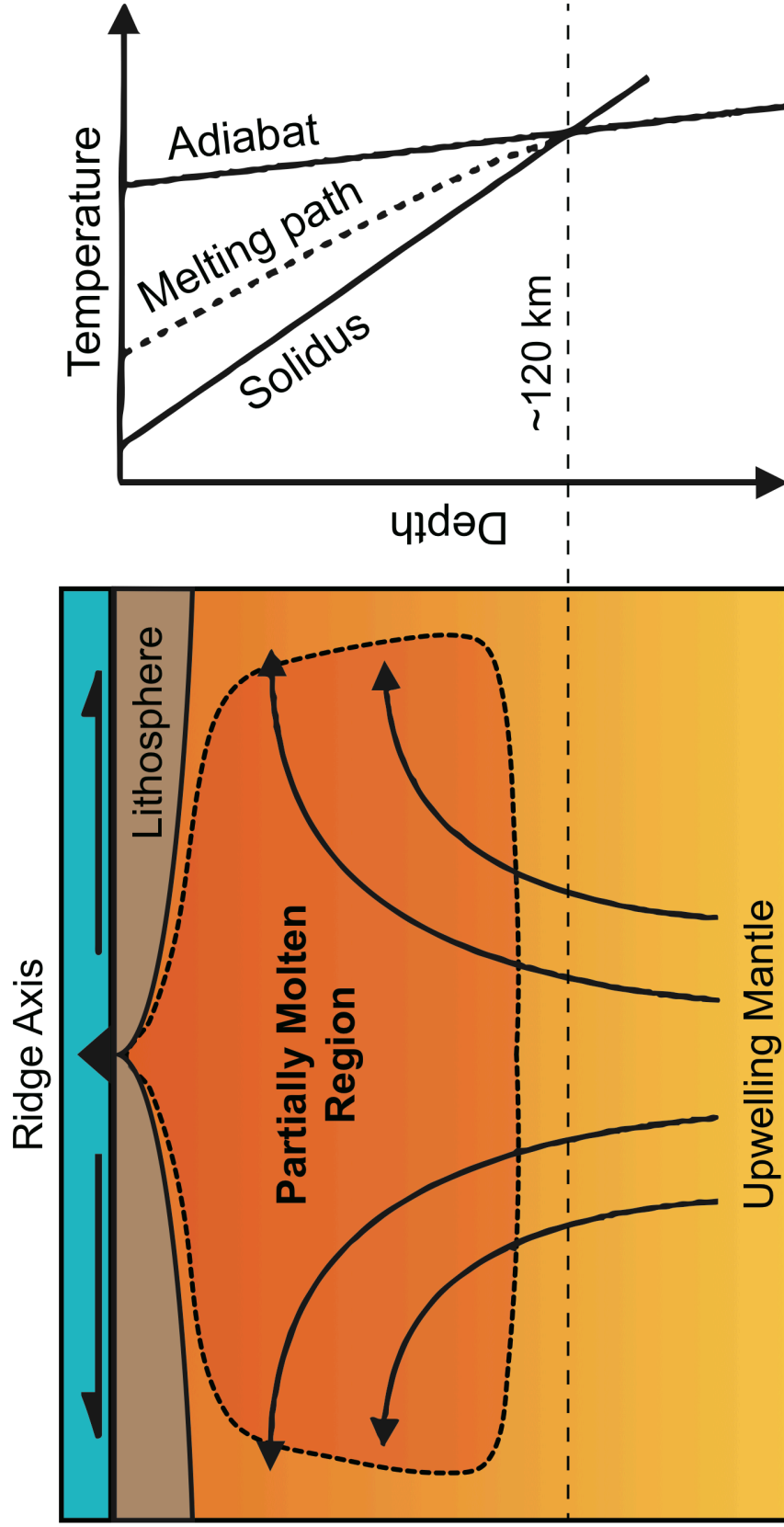


Figure 1.1: (Left) Schematic diagram of a mid-ocean ridge system. Light orange region represents mantle. Dark orange / red represents partially molten region. Blue region represents the lithosphere. Blue region is the ocean. Curved arrows in mantle illustrate flow direction of solid mantle material. Straight half arrows illustrate the spreading direction. Black triangle is the ridge axis. (Right) Temperature-depth profile of the upper mantle that illustrates the melting region in pressure-temperature space. Modified from (Weatherley, 2012).

gradient that induces decompression melting is also responsible for focusing melt (Phipps-Morgan, 1987; Ribe, 1988; Spiegelman and McKenzie, 1987). However, it was determined that divergence of the plates alone required unrealistic upper mantle viscosity values to account for magma ascent rates inferred from uranium-series data. Advection of trapped melt by means of buoyancy-driven convection (Buck and Su, 1989; Rabinowicz et al., 1984; Scott and Stevenson, 1989) was another popular theory at that time but required higher melt fractions than are observed by geophysical methods and a lower permeability than standard estimates.

Current thinking is that melt percolates through the upper mantle via porous flow along a grain boundary network of interstitial melt. Though porous flow is most often thought of in the context of fluid transport in the crust, where overburden pressures are sufficiently low to maintain interconnected pore space, the compressibility of melt is low enough to support an intergranular, interconnected network. The permeability of this network has been a parameter of high interest, since it relates the percolation velocity of melt on the aggregate-scale to local pressure gradients.

A number of attempts have been made to determine the permeability of partially molten rock, including consideration of idealized melt geometries, network modeling, and direct measurement on analogue systems. However, as will be discussed in more detail, these methods neither consider the proper three-dimensional (3-D) pore structure of partially molten rocks nor the correct chemistry.

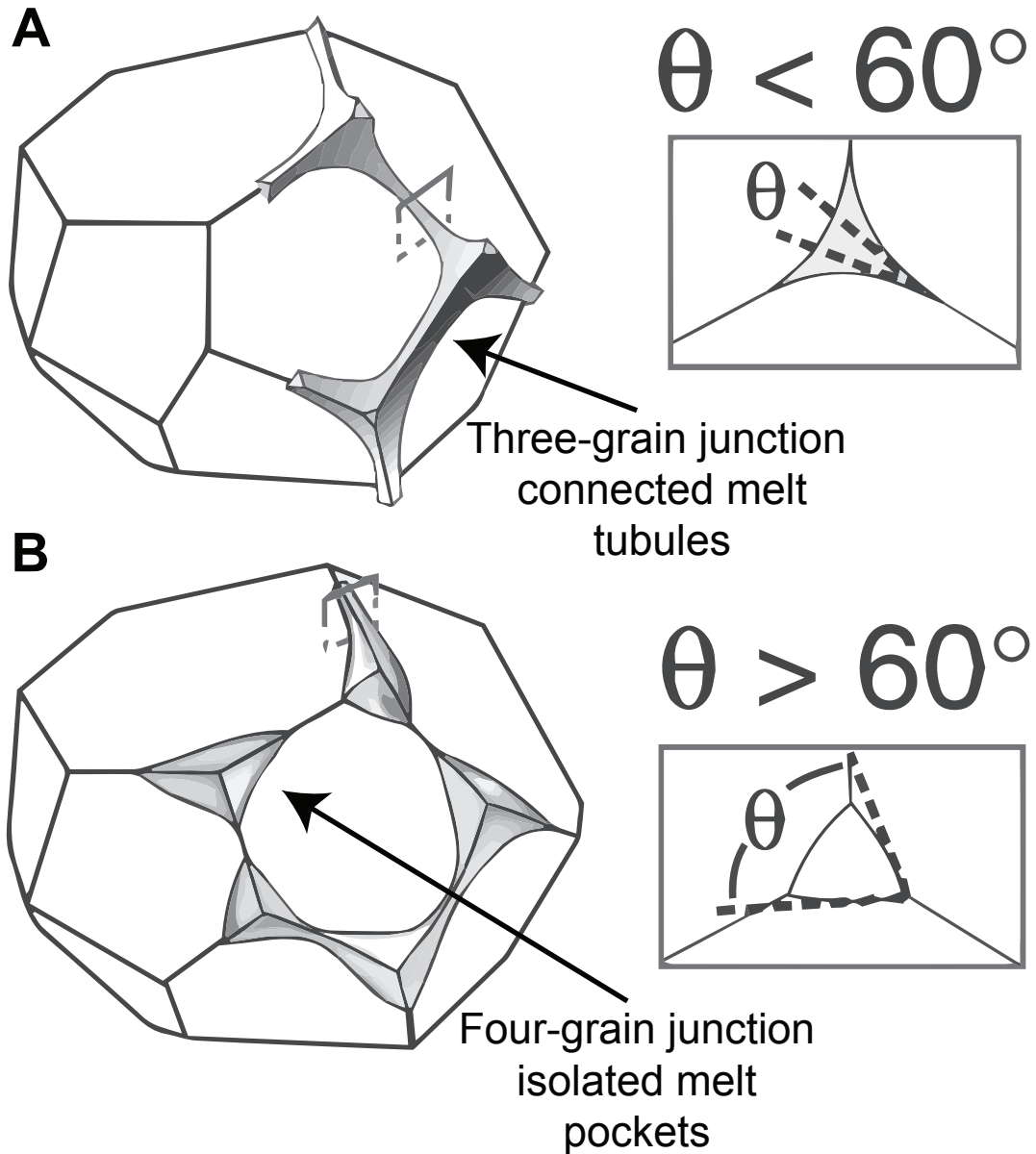


Figure 1.2: Idealized representation of three-dimensional melt geometry using isotropic tetrakaidecahedral grain shape. Included are two-dimensional cross-sections of melt features. (A) Interconnected melt tubules that form at three and four-grain junctions for $\theta < 60^\circ$. (B) Isolated melt tubules form at four-grain junctions for $\theta > 60^\circ$.

1.2 Melt microstructure

The grain-scale geometry of a partially molten rock is controlled by energy minimization processes (Bulau et al., 1979). For a melt fraction (ϕ) below the disaggregation limit ($\phi < 0.25$) (Scott and Kohlstedt, 2006), spatial variations in surface energy associated with interphase boundaries constitute thermodynamic gradients that drive melt into an equilibrium configuration (Smith, 1964; Waff and Bulau, 1979). The fluid transport, electrical, and mechanical properties of the rock depend crucially on the morphology and topology of the interstitial melt network.

A good indicator of the microstructure geometry is the dihedral angle (θ) (Smith, 1964, 1948), which is the angle that subtends two solid-melt interfaces. In general, θ varies from grain contact to grain contact, depending on the relative surface energy densities of the adjacent phase boundaries. However, for two identical, adjacent, isotropic grains separated by melt, θ is defined by the following relation:

$$\cos\left(\frac{\theta}{2}\right) = \frac{\gamma_{ss}}{2\gamma_{sm}} \quad (1.1)$$

where γ_{ss} and γ_{sl} are the surface energy densities of the solid-solid and solid-liquid phase boundaries, respectively. For $\theta < 60^\circ$ and any melt fraction, melt forms an interconnected network along triple junctions consisting of prismatic melt tubules that are connected at four-grain junctions (Fig. 1.2A) (von Bargen and Waff, 1986). Conversely, for $\theta > 60^\circ$, melt forms isolated pockets at grain corners (Fig. 1.2B) unless a critical melt fraction is exceeded. γ_{ss} and γ_{sl} are fundamental to the chemistry and mineralogy on either side of the interface. An aggregate composed of olivine, which is the primary upper mantle mineral component, and basaltic melt exhibits a median dihedral angle of $\sim 35^\circ$ (Waff and Bulau, 1982), so an olivine-basaltic melt

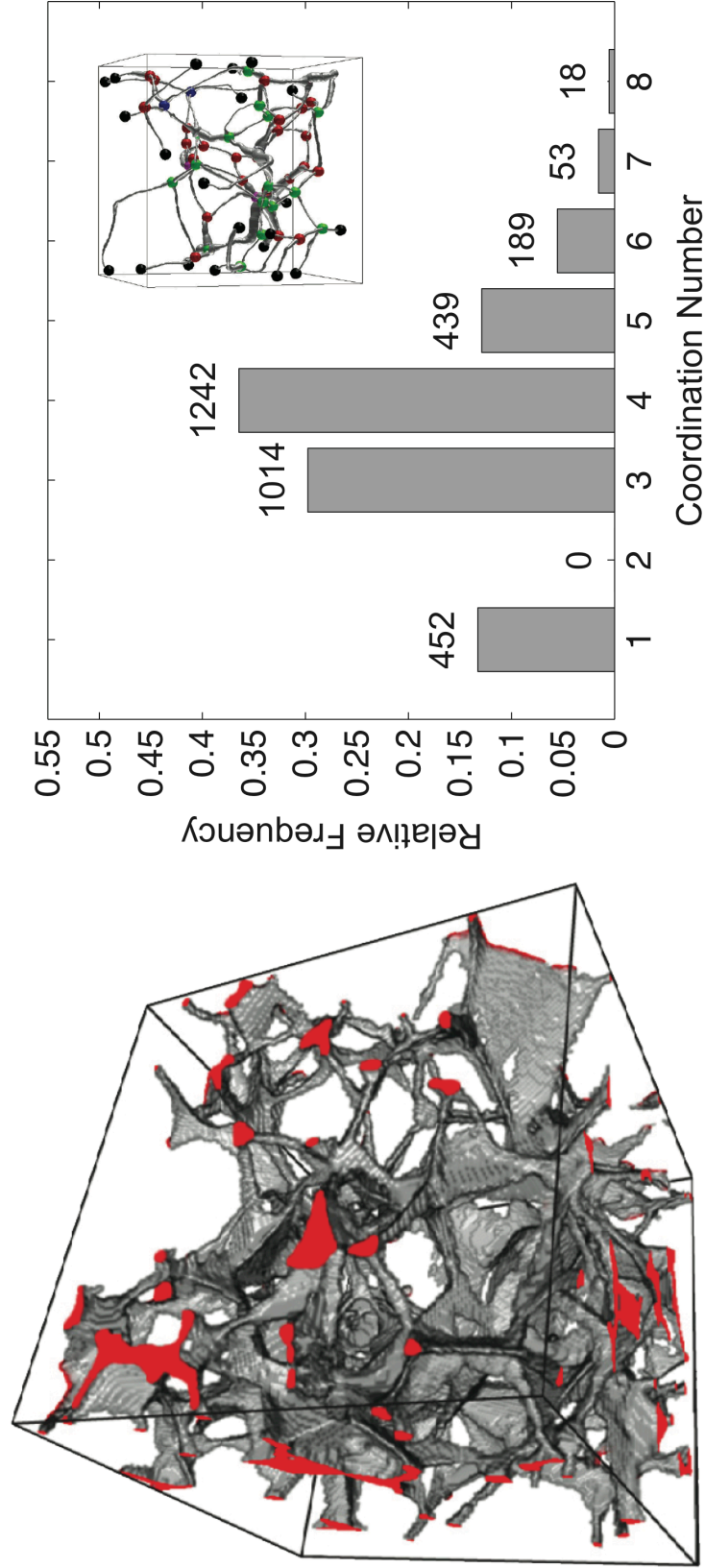


Figure 1.3: Demonstration of heterogeneous grain-scale melt distribution for subvolume cropped from an olivine-basaltic melt whose melt fraction ($\phi \sim 0.05$). Modified from Miller et al. (2014). (A) Volume-rendering of $140 \times 140 \times 140 \mu\text{m}^3$ melt microstructure (gray). Red highlights the intersection of the melt and the bounding box. (B) Histogram of coordination number computed for the simplified melt microstructure (insert).

aggregate should support an interconnected melt network.

Analysis of 2-D cross-sections (e.g. Cooper and Kohlstedt, 1982; Waff and Bulau, 1982; Cmiral et al., 1998; Faul and Fitz Gerald, 1999) reveal a range of melt features. Most those features are prismatic melt tubes that reside at three and four-grain junctions, which is consistent with the von Bagen and Waff (1986) model. However, additional melt features, such as melt films and melt pools, also exist due to the anisotropic surface energy density of olivine-basaltic melt interfaces (Faul, 2000; Laporte and Provost, 2000). This observation is confirmed by 3-D analysis of the coordination number distribution (Fig. 1.3), where the coordination number is defined as the number of melt features that connect at a single point and is a measure of the melt network topology. Fig. 1.3 highlights the diversity of features present in olivine-basalt aggregates (e.g. prismatic tubules, melt films, melt pools).

1.3 Permeability of the melt microstructure

1.3.1 Permeability of idealized geometries

An interconnected, interstitial melt network facilitates melt transport over distances larger than the grain-scale (Turcotte and Schubert, 2014). A crucial parameter used for modeling melt transport in mid-ocean ridge systems is permeability (k), which is a measure of the capacity of the rock to transport melt. Permeability is a power law function of melt fraction,

$$k = \frac{\phi^n d^2}{C} \quad (1.2)$$

where ϕ is the melt fraction and d is the average grain size [m^2]. C and n are power law parameters that depend on the morphology and topology of the melt network. For

idealized melt network geometries (e.g. Frank, 1968; von Bagen and Waff, 1986), Eqn. (1.2) can be analytically derived. For example, a network composed of uniform tetrakaidecahedral grains, i.e. prismatic melt tubules residing at three and connecting at four-grain junctions, permeability is given by Eqn. (1.2), where $n = 2$ and C is 1600 (von Bagen and Waff, 1986). Another model (Faul et al., 1994) that assumes ellipsoidal inclusions, an approximation to a partial melt with wet grain faces, yields a power law exponent of $n = 3$. However, melt geometries of real partially molten rocks are heterogeneous and exhibit a range of melt features at different melt fractions, in which case Eqn. (1.2) is an empirical relation.

In order to assess the influence of melt network heterogeneity on permeability, Zhu and Hirth (2003) used a network permeability model to randomly vary the diameter of melt tubules in a pack of isotropic, tetrakaidecahedral grains. They found that for a uniform tubule diameter, the permeability-melt fraction power law was the same as that analytically derived by von Bagen and Waff (1986). Though for randomly varying melt tubule diameters, computed permeabilities adhered to a power law exponent $n = 3$. Though a major step forward from idealized geometries, a systematic laboratory quantification of partially molten mantle rock permeability is needed.

1.3.2 Experimental constraints on permeability

Permeability is technically challenging to measure for partially molten systems because of the extreme pressure-temperature conditions required for melting and the high viscosity of basaltic melt. Therefore, a number of studies (e.g. Holness

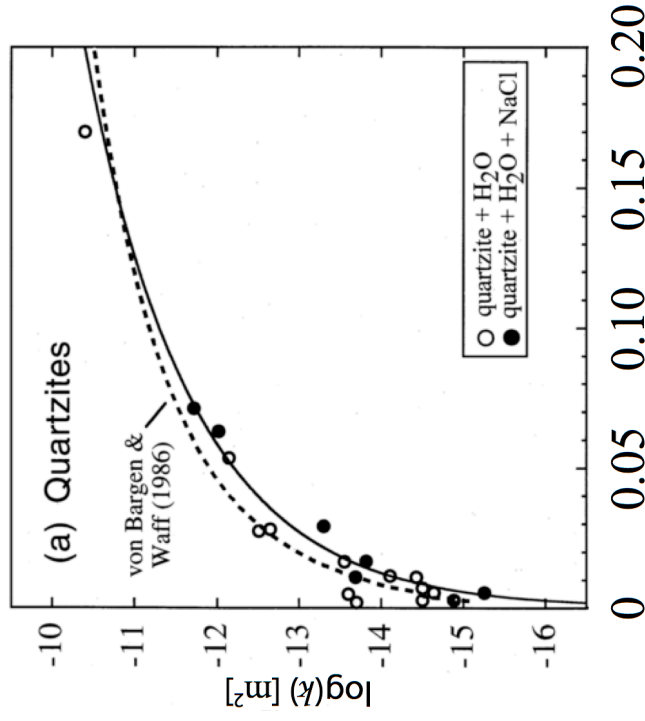
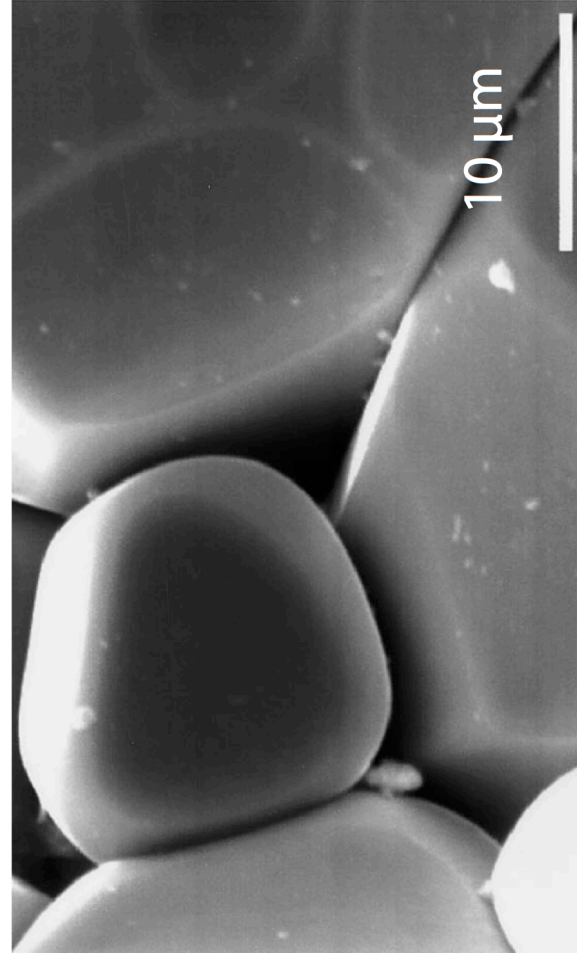


Figure 1.4: (Left) SEM image of a drained quartzite grain pack. (Right) Solid line is fit to permeability data of quartzite grain pack. Dotted line is expected permeability-porosity curve for idealized tubular melt geometry.

and Graham, 1991; Jurewicz and Watson, 1984; Mibe et al., 1998; Wark and Watson, 1998; Wark et al., 2003; Watson and Brenan, 1987) look to analogue systems that have wetting properties similar to the olivine-basaltic melt system. For example, Wark and Watson (1998) measured the permeability of aggregates composed of quartz plus H₂O brine ($\theta = 32^\circ$). They found permeability adheres closely to Eqn. (1.2), where power law parameters $n = 3$ and $C = 200$ (Fig. 1.4). Studies that used analogue materials provided valuable insight to the grain-scale fluid distribution in real, heterogeneous porous rocks. However, grain-scale fluid distribution is sensitive to distribution of surface energy – and therefore mineralogy and fluid chemistry – so it is unclear if the findings of analogue studies apply to partially molten mantle rocks, which are composed primarily of olivine. In order to properly constrain the permeability of partially molten mantle rock, a chemistry and mineralogy that is representative of the mantle must be used.

Several attempts to measure the permeability of olivine-basalts have been made. For example, Renner et al. (2003) measured the compaction rate of olivine-basaltic melt samples, undergoing draining in response to an imposed pressure gradient. By relating the measured compaction rate to permeability, they found their results implied a permeability-melt fraction relationship that qualitatively resembled Wark and Watson (1998) (power law parameters $n = 3$ and $C = 200$), but a rigorous fit to the data was not conducted. Furthermore, permeability is a property of the instantaneous melt geometry. As the melt fraction and grain-scale melt distribution changes during compaction, so does the permeability.

An additional experimental constraint on olivine-basaltic melt permeability-

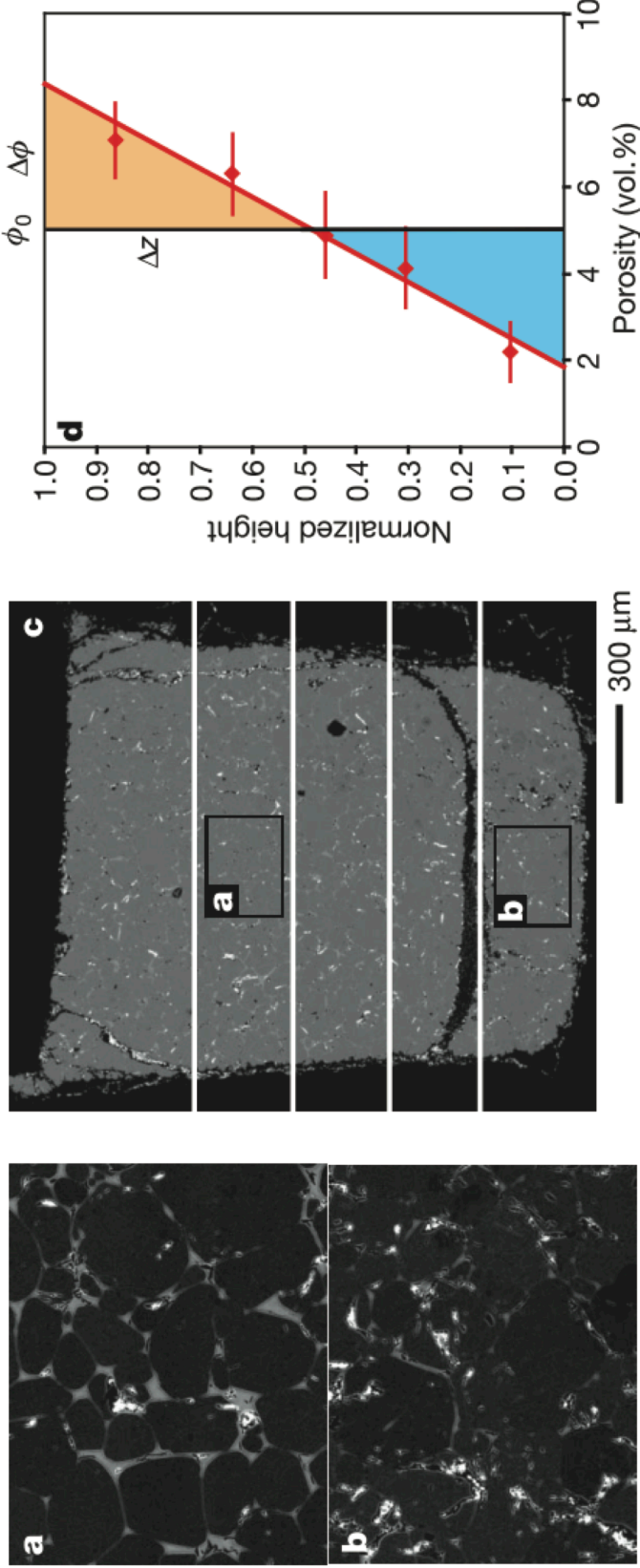


Figure 1.5: Melt migration in partially molten laboratory sample (Connolly et al., 2009). (A-C) Electron backscatter image of partially molten melt microstructure. (C) The sample is partitioned into five regions. In each region the average melt fraction is measured. (D) Measured melt fraction for each region in (C). Orange and blue regions represent an excess and deficit melt fraction relative to the nominal melt fraction.

melt fraction relationship was provided by Connolly et al. (2009), who used a high-temperature, high-pressure centrifuge to mimic compaction-driven flow that occurs during melt transport under upper mantle conditions. Their samples were spun to accelerations of 400-700 g, which greatly enhanced the rate of melt flow. Using a scanning electron microscope (SEM) to measure the porosity profile of their samples (Fig. 1.5) before and after centrifuging, they were able to back out sample permeability. Their sample was composed of olivine-basalt and had melt fractions of 0.05. Their data suggested a piece-wise permeability-melt fraction relationship: a quadratic ($n = 2$) dependence for low melt fractions and a cubic ($n = 3$) dependence on melt fraction for higher melt fractions, which they interpret as indicative of a change in melt morphology from a tubule-dominant network at low melt fraction to one that is populated by higher-coordination number connections. They estimated the geometrical constant C to range between 3 and 27, which is consistent with a highly heterogeneous grain-scale melt distribution. Though their experiment was a significant leap forward in linking permeability to the melt microstructure, it is not straightforward to assess boundary effects of their experimental setup. Therefore, it is necessary to independently constrain the permeability as a function of melt fraction using alternative methodology.

1.3.3 Electrical conductivity of partially molten mantle rocks

The electrical conductivity of partially molten mantle rock can be used as a tool for probing melt content of the mantle and for inferring the 3-D grain-scale distribution of melt in partially molten rock samples. The presence of partial melt

increases the electrical conductivity of mantle rock by several orders of magnitude (e.g. Roberts and Tyburczy, 1999; ten Grotenhuis et al., 2005; Yoshino et al., 2010). For an olivine-basaltic melt aggregate, the bulk electrical conductivity is an average of the electrical conductivities of olivine and basaltic melt, which is on the order of 0.01 S/m (Constable, 2006; Yoshino et al., 2010) and 1-10 S/m (Roberts and Tyburczy, 1999; ten Grotenhuis et al., 2005; Yoshino et al., 2010) for olivine and basaltic melt, respectively. If melt exists as isolated pockets, i.e. melt fraction is below the percolation threshold and dihedral angle is greater than 60°, olivine and melt will conduct in series. As a result to bulk electrical conductivity of the aggregate be very low. Conversely, if melt forms an interconnected network, bulk electrical conductivity is high, since melt and olivine conduct electricity in parallel. The bulk electrical conductivity of an actual partially molten rock will be somewhere between these two end-member cases.

Much like permeability, bulk electrical conductivity of partially molten rocks adheres to a power law, specifically Archie's Law:

$$\sigma_{\text{bulk}} = A\sigma_{\text{melt}}\phi^m \quad (1.3)$$

where A and m are power law parameters, σ_{bulk} and σ_{melt} are the electrical conductivities of the bulk and melt phase, and ϕ is melt fraction. Eqn. (1.3) is an empirical relation that assumes the mineral phase is a good insulator relative to the melt phase, which is true for olivine-basaltic melt aggregates.

Note the similarities between Eqn. 1.2 and Eqn. 1.3. Both are power laws that relate a bulk transport properties to characteristics of the melt microstructure. The fact that the same pathways that facilitate fluid flow are also available to conduction of

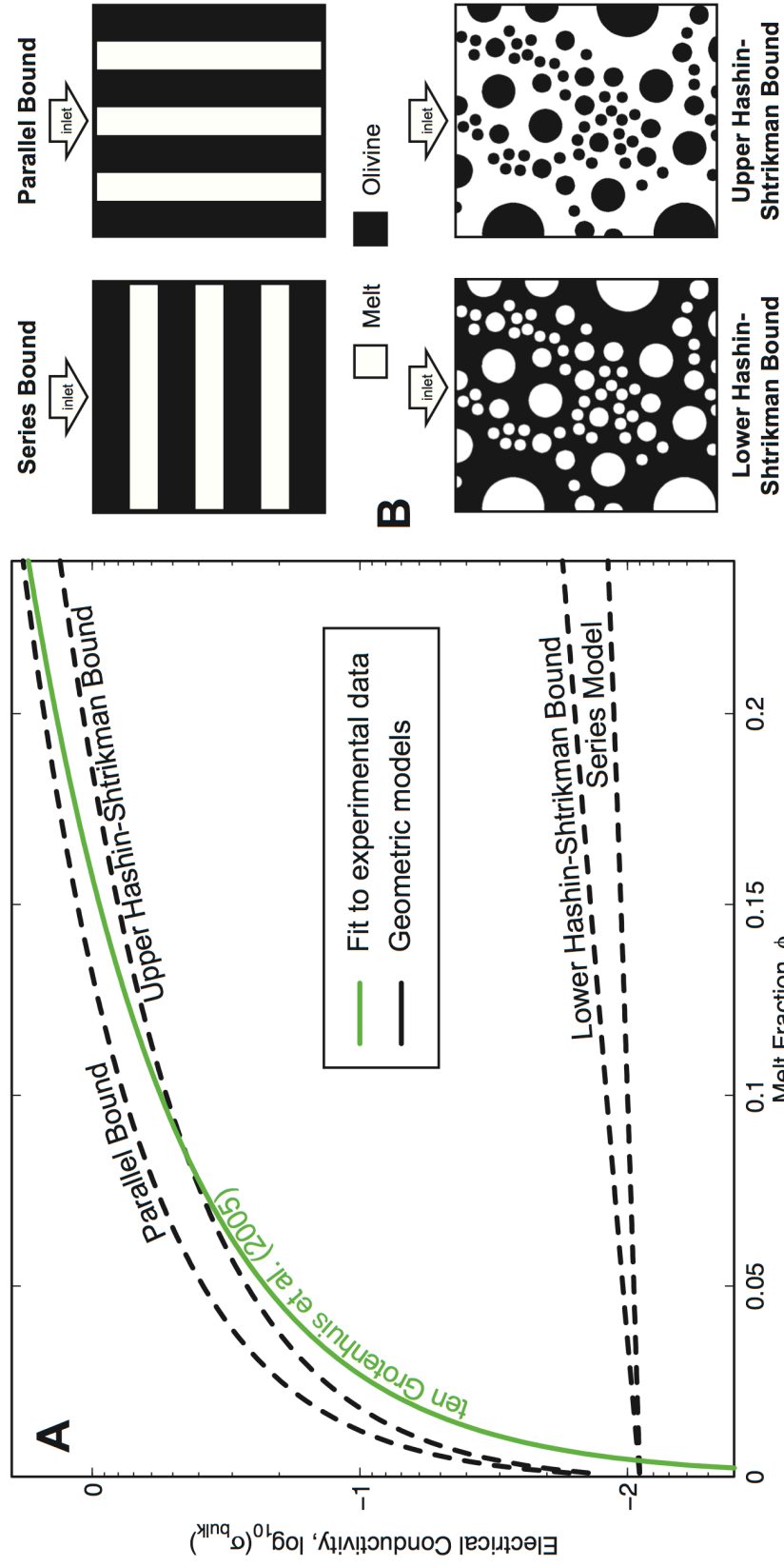


Figure 1.6: Interpretation of mixing models. (A) Comparison of experimentally obtained bulk electrical conductivity-melt fraction (Archie) relation and geometric mixing models. (B) Illustrations of literal interpretations of geometric mixing models: conduction through series/parallel layers (series/parallel bounds) and isolated and connected spherical melt layers (lower-upper Hashin-Shtrikman bound).

electrical current has garnered significant interest in linking permeability and electrical conductivity. We address the possibility of using electrical conductivity as a tool for inferring permeability in Chapter 3.

1.3.4 Using electrical conductivity to infer 3-D melt distribution

It is common practice to infer 3-D melt geometries from measured values of σ_{bulk} . Fig. 1.6 is comparison of an Archie relation for the olivine-basaltic melt system, obtained by fitting measured bulk electrical conductivity data, and the geometric mixing models (Fig. 1.6A). Measured values of σ_{bulk} for olivine-basaltic melt aggregates appear to straddle the Hashin-Shtrikman upper bound, which assumes a non-uniform pack of spherical grains completely wetted by a uniform layer of melt (Fig. 1.6B). However, this interpretation is inconsistent with microscopy studies that observe coexisting melt tubules, melt films, and melt pools. A derivation of electrical conductivity for a real partially molten rock geometry from microstructural considerations, which is discussed in Chapter 3, is therefore needed to explain the high bulk electrical conductivities observed in synthetic partial melts.

1.3.5 3-D melt distribution from X-ray micro-computed tomography

Rather than inferring a 3-D melt distribution of olivine-basaltic melt samples by comparing measured bulk properties to idealized mixing models, the three-dimensional melt microstructure can be obtained using synchrotron X-ray micro-computed tomography (μ -CT) (Zhu et al., 2011). μ -CT is a three-dimensional imaging technology that exploits the difference in relative X-ray absorption

efficiencies of materials. μ -CT has been used for decades to study the pore structure of crustal rock samples. However, μ -CT has only recently been applied to study the olivine-basalt partially molten system in part due to the technical challenge associated with resolving the small density contrast ($\sim 400 \text{ kg m}^{-3}$) between olivine and basaltic. Novel reconstruction algorithms that incorporate diffraction-enhanced tomography (Fitzgerald, 2000), also known as qualitative phase retrieval, allow one to highlight grain-basalt interfaces. The resulting high-resolution, 3-D image constitutes a virtual rock sample on which microstructural analysis or digital rock physics (DRP) experiments can be conducted.

1.4 Basics of X-ray micro-computed tomography

There are two categories of X-ray μ -CT: absorption-contrast and phase-contrast tomography. Both are inverse problems that are solved using some implementation of the filtered-back-projection method (see Kak and Slaney (1988) for a review of principles).

Absorption-contrast tomography utilizes spatial variations in the density distribution to image an object. The estimated spatial density distribution can be computed by inverting a series of projections taken along different ray paths through the object (Fig. 1.7 & Fig. 8). Each projection contains a record of the proportion of X-ray attenuation integrated along the ray path. For each X-ray path, the X-ray intensity I is given by

$$I = I_0 \exp \left[- \int_{-\infty}^{+\infty} \mu(x) dx \right] \quad (1.4)$$

where I_0 is the intensity of the incident X-ray and μ is the absorption coefficient along

the path of the X-ray. Reconstructing the 3-D image amounts to finding the absorption coefficients μ that relate the known incident X-ray intensity to the attenuated signal recorded in the projections. Robustly resolving material interfaces using absorption contrast tomography requires a sufficiently large density contrast between materials. As was discussed briefly in Section 1.3.4, the density contrast between olivine and basalt is too low for absorption-contrast alone to be effective at imaging samples composed of olivine and quenched basaltic melt.

Additional information can be obtained from the diffraction signal embedded in the X-ray projection. There are two classes of phase-contrast tomography: “qualitative” phase-contrast tomography (e.g. Fitzgerald (2000)), commonly referred to as “edge-enhancement,” incorporates information about diffraction of X-rays at the mineral-mineral or mineral-fluid interface to highlight those interfaces. Qualitative phase-contrast tomography was successfully applied to monomineralic partially molten forsterite-basalts (Zhu et al., 2011). However, even edge-enhancement is not sufficient to resolve polymineralic aggregates that contain olivine-orthopyroxene (opx) and opx-basalt interfaces, whose density contrast is $\sim 70 \text{ kg m}^{-3}$. Therefore, quantitative phase-contrast-tomography (e.g. Paganin et al., 2002), which exploits the spatial distribution of the index refraction, can be used to improve the image quality. Common quantitative phase retrieval algorithms (Paganin et al., 2002) essentially perform joint-inversions between absorption-contrast and phase-contrast tomography. They provide excellent image quality when the density contrast between components is small.

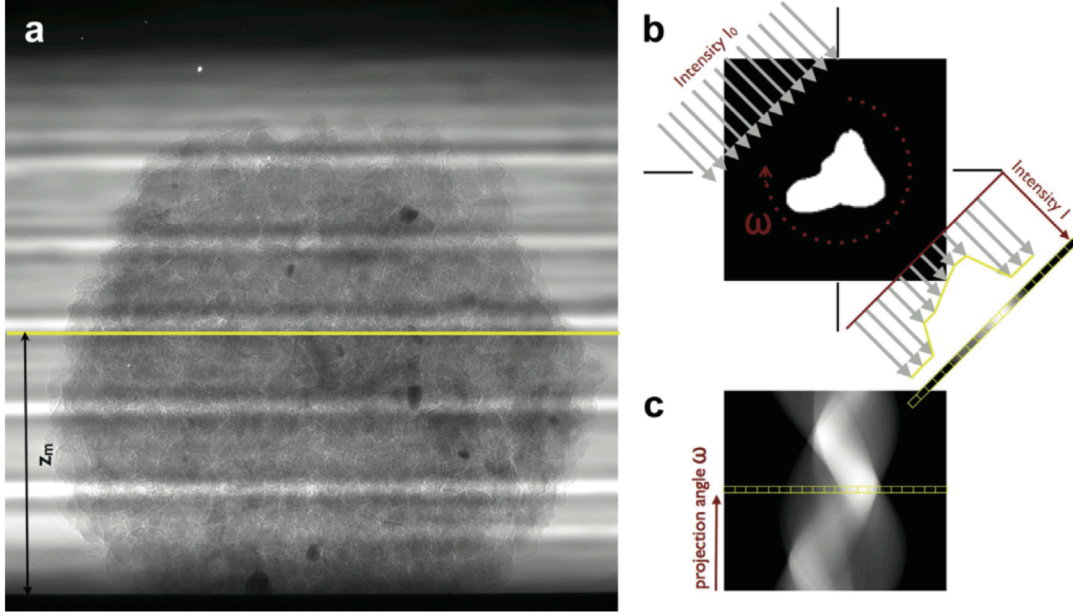


Figure 1.7: Schematic of X-ray reconstruction process (Fusseis et al., 2014). (a) X-ray radiograph. (b) Schematic demonstrating the collection of X-ray radiographs for different angles. (c) Transformation of radiographs into sinogram.

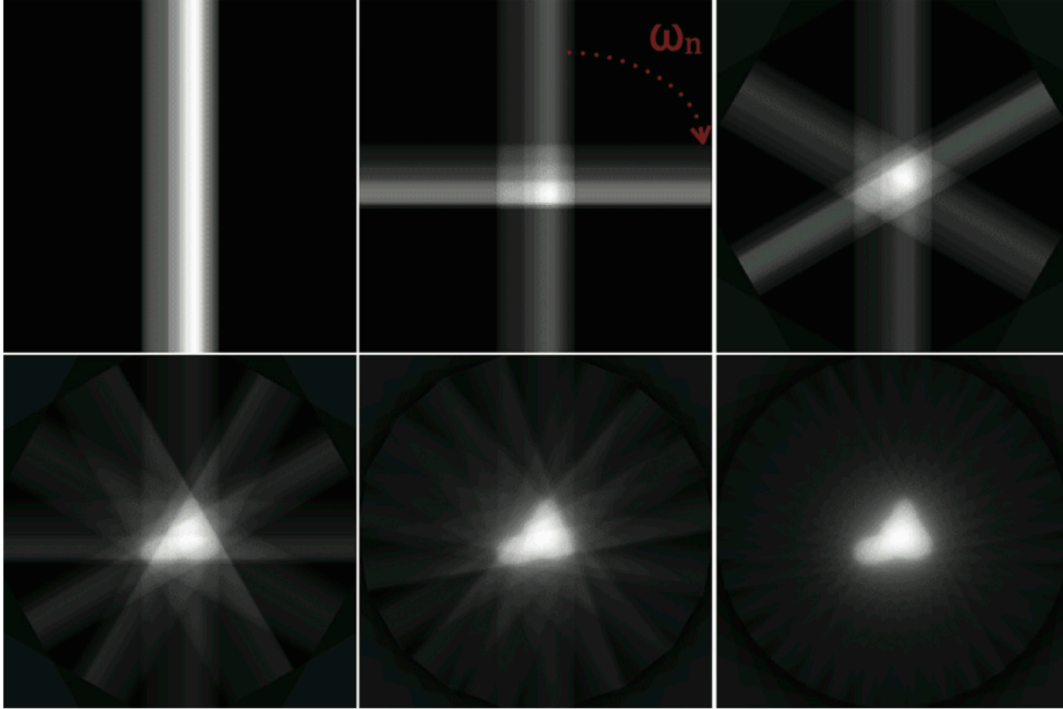


Figure 1.8: Demonstration of image reconstruction using a single cross-sectional slice (Fusseis et al., 2014). (Top) Illustration of ray paths whose grayscale value corresponds to the grayscale value of the projection. (Bottom) Reconstruction performed using the number of projections in the corresponding above image. The number of projections used in reconstruction increases from left to right. As the number of projections increases, the object being imaged becomes better resolved, i.e. closer to Fig. 1.7b.

1.6 Segmentation of grayscale image

The grayscale output of the reconstruction algorithm is not immediately lendable to automatic quantification and numerical computation of physical properties. An additional step, often referred to as segmentation, needs to be conducted to transform the grayscale image to a label image, wherein each voxel is assigned a non-negative integer identification number. Once “segmented,” a number of algorithms can be used to quantify the morphology and topology of the mineral and fluid structure. Label images can be easily discretized and be used as the computational domain in digital rock physics (DRP) simulations to compute physical rock properties.

The goal of segmentation is to accurately capture the spatial distribution of materials based on their grayscale values so that the reconstructed digital rocks are representative of the real samples. Refer to Fousseis et al. (2014) for a quantitative comparison of segmentation techniques applied to X-ray μ -CT of geological materials. For the purpose of this project, it suffices to say that common segmentation algorithms fall into two categories: global and local. Global segmentation algorithms involve thresholding the data by a globally defined variable, such as the grayscale value at the inflection of the grayscale histogram computed for the whole subvolume. Conversely, local segmentation algorithms assign label identifiers to pixels based on locally varying quantities. Local segmentation algorithms are better at repressing the random or speckled noise and the long-wavelength grayscale variations but are computationally expensive to perform on large volumes.

1.7 Digital rock physics applied to partially molten rock: an overview

The true power of μ -CT comes to light when it is combined with numerical simulations. The 3-D label images constitute virtual rock samples on which any virtual rock physics experiment can be conducted and any range of parameters can be tested with relative ease. With a DRP approach, tweaking an experiment amounts changing input parameters or boundary conditions. Material properties can be accurately derived from first principles and directly linked to characteristics of the rock microstructure.

I used DRP techniques to characterize microstructure and compute permeability and electrical conductivity of olivine-basalt aggregates (Chapter 2 and 3) and olivine-opx-basalt (Chapter 4 and 5) aggregates as a function of melt fraction. Melt morphologies and topologies are quantified on statistically representative volumes and linked to transport properties. A number of 3-D image processing, segmentation, and automated quantification tools are also discussed.

Chapter 2: Experimental quantification of permeability of partially molten mantle rocks

Abstract

Melt percolation in mantle rocks is currently poorly constrained, especially at low melt fractions. At mid-ocean ridges, for example, geochemical and geophysical observations produce divergent estimates of how much melt is present in the mantle and how quickly it moves. Accurate estimates of permeability and grain-scale melt distribution in mantle rock are necessary to reconcile these observations. We present three-dimensional (3-D), 700 nm-resolution images of olivine-basalt aggregates, containing nominal melt fractions (ϕ_n) between 0.02 and 0.20. Samples were prepared from a powdered mixture of San Carlos olivine and high-alumina basalt and hot-pressed in a solid-media piston-cylinder apparatus at 1350 °C and 1.5 GPa. Images were obtained using synchrotron X-ray micro-computed tomography (μ -CT) from the Advance Photon Source at Argonne National Laboratory. Stokes flow simulations, conducted using the digital melt volume as the numerical domain, determine that the permeabilities of experimental charges range from 2×10^{-16} to $5 \times 10^{-13} \text{ m}^2$ for $\phi_n=0.02$ to 0.20, respectively. The simulation results are well represented by the power-law relation between permeability (k) and melt fraction (ϕ), $k = \phi^n d^2 / C$, where $n = 2.6 \pm 0.2$, and assuming a grain size of 35 μm , $C = 58^{+36}_{-22}$. These results place important new constraints on rates of melt migration and melt extraction within partially molten regions of the mantle.

2.1 Introduction

At mid-ocean ridges, the divergence of lithospheric plates causes an upwelling of hot mantle. The pressure relief during ascent carries peridotite across its solidus and induces partial melting. The melt, which is less dense than the surrounding mantle, separates from the solid and percolates towards the surface via porous and possibly channelized flow (e.g. Kelemen et al., 1997). The melt extraction rate is governed by the permeability of the mantle, which is highly influenced by the amount of melt present as well as the topology and connectivity of the melt network. Despite its importance for understanding melt transport in the mantle, the permeability of partially molten mantle rock is poorly constrained. The aim of this study is to provide better permeability estimates through the quantification of grain-scale melt distribution.

At textural equilibrium, the relationship between permeability and the grain-scale melt distribution in a partially molten rock takes the form of a power law (Cheadle, 1989; Connolly et al., 2009; McKenzie, 1984; Ricard et al., 2001; von Bargen and Waff, 1986; Wark and Watson, 1998),

$$k = \frac{\phi^n d^2}{C} \quad (2.1)$$

where d is grain size, n is the power law exponent, and C is a geometric factor influenced by the dihedral angle. For an isotropic system with uniform grain size and shape, $n=2$ (McKenzie, 2000; von Bargen and Waff, 1986). However, for more complex systems, where the effects of crystal anisotropy and grain-scale heterogeneity are no longer negligible, higher values of n should be used. For example, a value of $n=3$ represents well porous flow through a non-uniform network of packed

tetrakaidekahedral grains (Zhu and Hirth, 2003). These model results have been corroborated by permeability experiments conducted on analogue systems composed of quartzite + H₂O and calcite + H₂O where grain size distribution is non-uniform, grain shapes are anisotropic, and $n \sim 3$ (Wark and Watson, 1998).

Mineralogy plays an important role, through its influence on surface free energy, in determining the minimum-energy configuration of the system. Therefore, experiments conducted on partial melts with chemistry similar to the mantle must be considered. Some permeability experiments (Connolly et al., 2009; Renner et al., 2003) have been conducted for olivine partial melts. They find that the permeability of partially molten olivine basalt at high melt fractions ($\phi > 0.02$) is consistent with a power law where $n \sim 3$. However, permeametry of partially molten aggregates in these experiments is technically challenging. Consequently, the results of such studies are subject to considerable uncertainty.

Grain-scale melt distribution is typically studied by examining backscattered electron images from two-dimensional (2-D) cross-sections of isostatically pressed samples (e.g. Cmíral et al., 1998; Faul and Fitz Gerald, 1999). By assuming a model about the three-dimensional (3-D) connectivity of the melt network, it is possible to infer and estimate sample permeability using the 2-D data. However, those estimates are innately ambiguous, since permeability is an intrinsic property of the 3-D microstructure (Zhu et al., 2011). Therefore, a fully 3-D approach must be employed in order to accurately determine sample permeability. Two methods may be employed for characterizing microstructures in three dimension: serial cross-sectioning (Garapić

et al., 2013; Wark et al., 2003) and synchrotron X-ray micro-computed tomography (μ -CT) (Watson and Roberts, 2011; Zhu et al., 2011). This study focuses on the latter.

Constraints on mantle permeability come from both geochemical and geophysical observations. Analyses of uranium-series isotopes in mid-ocean ridge basalts (MORB) (Lundstrom et al., 1995; McKenzie, 2000, 1985; Sims et al., 2002; Stracke et al., 2006), have shown a measureable degree of secular disequilibrium between ^{238}U and its shorter-lived daughter nuclides ^{230}Th and ^{236}Ra . Preservation of secular disequilibrium at the surface implies a low melt fraction retained by the mantle, $\phi < 0.01$, with a relatively fast upwelling velocity at $\sim 1 \text{ m yr}^{-1}$ (Kelemen et al., 1997). By contrast, geophysical observations imply considerably higher melt fraction. For example, seismic and magnetotelluric data (Evans et al., 1999; The MELT Seismic Team, 1998) from the East Pacific Rise 17°S give evidence that the melt fraction in the mantle is 0.01 to 0.02, implying that melt extraction is inefficient at lower melt fractions. In a more recent study, Key et al. (2013) reported a melt fraction close to 0.10 under the East Pacific Rise 9°N using magnetotelluric inversions. Accurate estimates of permeability of partially molten rocks are needed to reconcile the apparent contradiction in melt fraction.

In this study, we utilize high-resolution μ -CT to digitally capture the 3-D melt distributions of olivine-basalt aggregates isostatically pressed in a piston-cylinder apparatus at 1350 °C and 1.5 GPa. Nominal melt fractions (ϕ_n) of samples systematically ranged from 0.2 to 0.20 (Zhu et al., 2011). To demonstrate textural equilibrium of these experimental charges, we also conducted time series experiments at nominal melt fraction of 0.05 (refer to Appendix A). For each sample, we selected

several representative subvolumes and characterized their permeability, grain size distribution, and melt interconnectivity. The permeability of each subvolume was calculated by numerically solving the Stokes fluid questions for the velocity and pressure fields within the digital melt microstructure. Permeability was plotted as a function of the measured melt fraction (ϕ_m) in the corresponding subvolume and an empirical relation between permeability and melt fraction was obtained. Our results provide new experimental constraints on the permeability and melt distribution of partially molten rocks.

2.2 Experimental Methods

2.2.1 Sample Preparation

Experimental charges were prepared from a powder mixture of natural, high-alumina basalt (Mg # = 0.705) and San Carlos olivine (\sim Fo₉₀) (Zhu et al., 2011). Olivine grains were sorted using a sieve to a maximum grain size of 10 μ m. The nominal melt fraction desired for each sample was obtained by varying the basalt content of the mixture, which was then homogenized with ethanol for six hour-long cycles in an automatic agate mortar and pestle. The homogenized mixtures were pressed into cylindrical pellets under a 1-ton press, placed into graphite capsules (Fig. 2.1A), and dried overnight at 400 °C to remove water. The whole assembly was centered in a straight-walled graphite furnace using crushable MgO spacers. The pressure medium for all experiments consisted of a CaF₂ sleeve.

Experiments were conducted using 1.27 cm assemblies (Boyd and England, 1960). Pressure was initially applied using the cold piston-in technique (Johannes et

al., 1971). The friction correction for the assemblies was calibrated against the Ca-tschermakite breakdown reaction at 1.2 to 1.4 GPa and 1300 °C (Hays, 1966) and determined to be less than the pressure uncertainty of the pressure gauge, so no correction has been applied to the reported pressures. Temperature was measured and controlled using a $W_3Re_{97}/W_{25}Re_{75}$ thermocouple; no correction for the effect of pressure on thermocouple EMF has been applied to the reported temperatures. N_2 was flowed over the thermocouple wires to minimize thermocouple oxidation over the course of an experiment. Temperatures are estimated to be accurate to $\pm 10^\circ C$ and pressures to ± 50 MPa. The temperature difference over the capsule was determined to be less than 5 °C using offset thermocouples. Experiments were terminated by shutting off the power. Upon completing each experimental run, the graphite capsule was sawed open to expose the surface of the experimental charge (Fig. 2.1B). The exposed surface was polished and reflected light photomicrographs were taken. A cylindrical ~ 0.9 mm diameter cylindrical samples was then cored from each charge to be used for μ -CT analysis (Fig. 2.1C).

Two suites of experiments were conducted (Table 2.1). The first suite was a time series, which was conducted to determine the minimum time required for a sample to reach textural equilibrium. All of the time series samples have a nominal melt fraction of 0.05 and the sintering time varied systematically from 42 to 336 hours (see Appendix A). The second suite of samples consisted of nominal melt fractions of 0.02, 0.05, 0.10, and 0.20. The sintering time for each sample was sufficiently long to ensure textural equilibrium (Zhu et al., 2011).

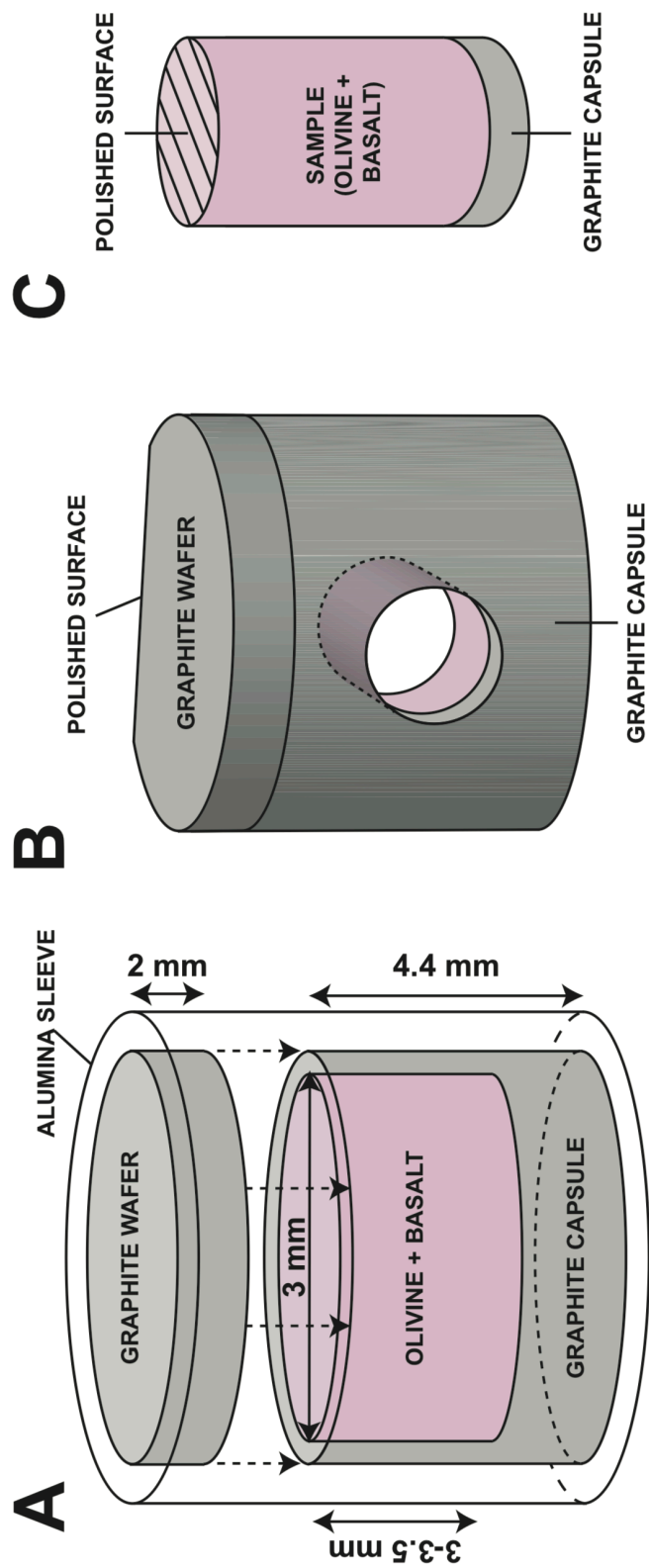


Figure 2.1: Schematic diagram of experimental setup inside the piston-cylinder apparatus. The sample assembly consists of the powdered starting material (pink) and the graphite holder (gray) in an alumina sleeve. Dimensions of the MgO tube, the sample assembly and the MgO rod are as marked.

Sample ID	Nominal Melt Fraction, ϕ_n	Sintering Time [hours]	Measured Melt Fraction ^a , ϕ_m	Grain Size ^b , d [μm]	Permeability ^c , k [m^2]
scoba-5	0.20	265	0.18 ± 0.02	$40 (-17 / +28)$	$2.3 (-0.4 / +0.4) \times 10^{-13}$
scoba-6	0.10	240	0.079 ± 0.009	$37 (-14 / +22)$	$1.9 (-0.5 / +0.6) \times 10^{-14}$
scoba-9	0.02	336	0.015 ± 0.003	$42 (-20 / +39)$	$4.1 (-0.7 / +0.8) \times 10^{-16}$
scoba-12 ^d	0.05	168	0.048 ± 0.004	$32 (-12 / +18)$	$5.2 (-1.1 / +1.3) \times 10^{-15}$
scoba-13 ^d	0.05	42	N/A	N/A	N/A
scoba-14 ^d	0.05	84	N/A	N/A	N/A
scoba-15 ^{d,e}	0.05	336	0.0570	29.6	7.7×10^{-15}

Table 2.1: ^a ϕ_m are arithmetic average measured melt fractions and 1σ standard deviations computed over range of subvolumes per sample. ^b d are geometric average equivalent diameters with 1σ standard deviations computed for EDD of the aggregated subvolumes. ^c k are geometric average permeabilities with 1σ geometric standard deviations computed over range of subvolumes per sample. ^dTime series experiments: no physical properties calculated on samples that have not yet achieved textural equilibrium. ^eThe values for scoba-15 were calculated from only one $350 \times 350 \times 350 \mu\text{m}^3$ subvolume.

2.3 Analytical Methods

2.3.1 Synchrotron X-ray micro-computed tomography

Microtomography was conducted at 2-BM of the Advanced Photon Source at Argonne National Laboratory, Argonne, IL. A multi-layer monochrometer was used to select a narrow band (27 keV) of X-rays. Those photons were then passed through the olivine-basalt sample (Fig. 2.2). On the opposite side of the sample, the X-rays were transmitted to a LuAg:Ce scintillator, converting them into visible light. A CCD camera was used to detect the visible light, and the light intensity was recorded. The sample was rotated 180° in 0.12° increments to build a digital volumetric representation of the sample in about 20 minutes (Fig. 2.2). For each sample, the raw intensity data was processed using GidRec (Dowd et al., 1999) into a stack of image slices. Each slice is a grayscale image whose constituent pixels have values that are functions of X-ray attenuation, which is in turn, a function of material density. In this way, μ -CT is used to differentiate phases, so long as the density contrast between the phases is substantial.

Silicate melt samples pose a unique problem in that the density contrast between olivine and basalt is not sufficient to differentiate the phases using standard phase contrast techniques. To circumvent this issue, we employed diffraction-enhanced imaging (Fitzgerald, 2000) to improve the contrast between olivine and basalt (Zhu et al., 2011). Diffraction-enhanced imaging utilizes the interference pattern, which occurs in the near-field Fresnel diffraction regime, to highlight the olivine-basalt interfaces and produce high-resolution 3-D microstructure of olivine-basalt systems.

IMAGING SCHEMATIC

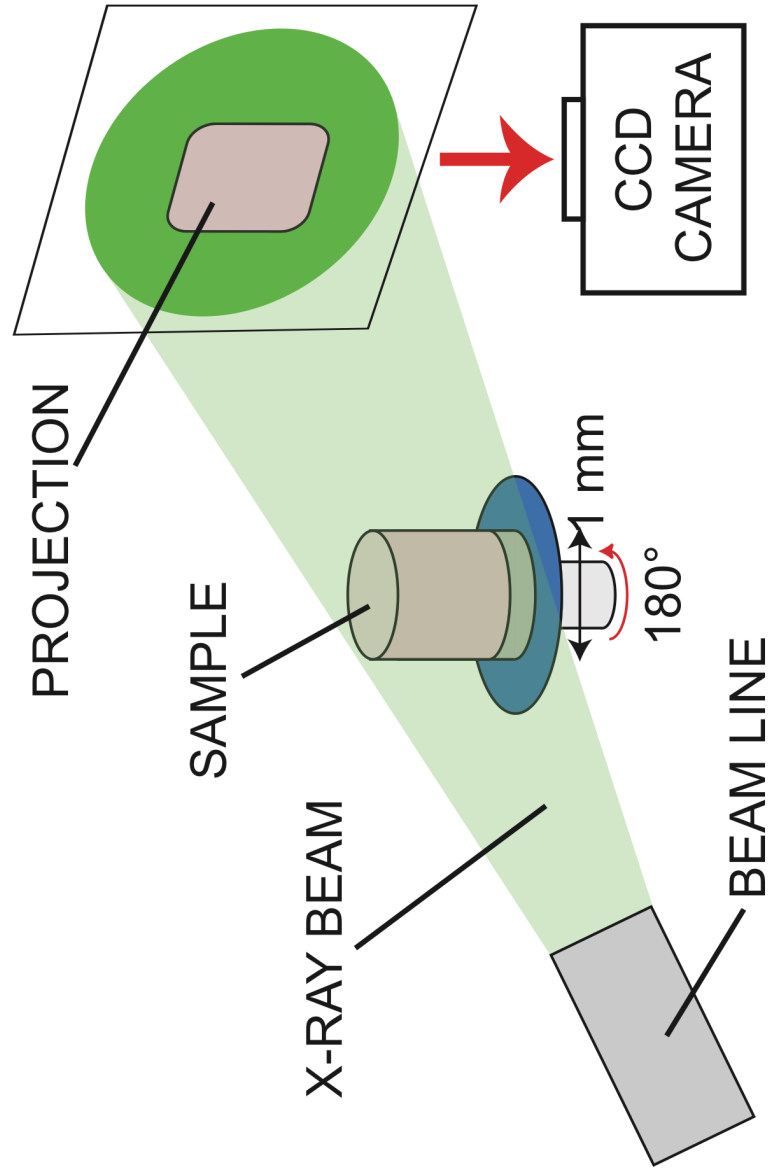


Figure 2.2: Schematic diagram of the imaging process. Sample is rotated through a mono-chromatic X-ray (green) beam. Incident light is converted from X-ray to visible light by a scintillator (white plane) so that it can be recorded by CCD camera. A projection is recorded at each angle increment for 180°. A radon transform is used to invert the projection data and obtain a digital, 3-D reconstruction of the whole sample.

2.3.2 Subvolume selection

Due to limited computation power, we selected only a few cubic subvolumes per sample for analysis. The size of those subvolumes ranged from $140 \times 140 \times 140 \mu\text{m}^3$ (i.e. $100 \times 100 \times 100 \text{ pixel}^3$) to $350 \times 350 \times 350 \mu\text{m}^3$ (i.e. the $500 \times 500 \times 500 \text{ pixel}^3$) (Fig. 2.3). We determined through a series of permeability analyses on progressively larger subvolumes that a $350 \times 350 \times 350 \mu\text{m}^3$ subvolume is sufficiently representative of the sample microstructure. Refer to Appendix A.1 for details.

Several $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes from each sample were analyzed. Although each subvolume is susceptible to local heterogeneities in the melt microstructure, taken together, these subvolumes adequately represent the melt microstructure of the entire sample. Analyses of sample permeability, grain size, and interconnectivity were conducted using a combination of Avizo[®] and Matlab[®] software.

2.3.3 Noise reduction and segmentation techniques

To reduce noise and suppress artifacts that remain from the imaging process, we employed a non-local means filter (Buades et al., 2005) and an anisotropic diffusion filter (Weickert et al., 1998) (Fig. A.1). Once we reduced the noise to an acceptable level, we implemented a series of algorithms to segment the grayscale data. Segmentation is a procedure by which we transform grayscale data into a binary label file required for our quantitative analyses of the microstructure (Fig. A.2). Two techniques were used for segmenting the grayscale data: a marker-based watershed transformation and a top-hat threshold.

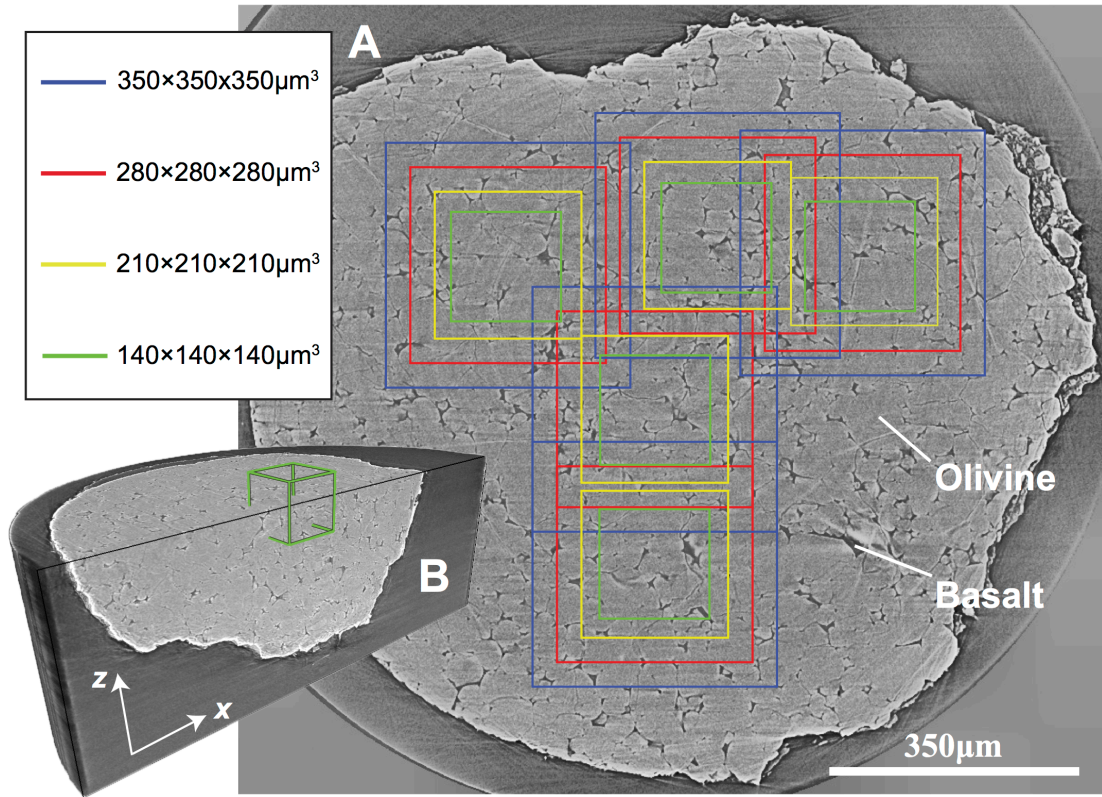


Figure 2.3: Subvolume sampling. (A) Tomography slice from the middle of scoba-12 ($\phi_n = 0.05$). Subvolumes of various sizes, shown as colored boxes, are defined from (B) the larger 3-D reconstructed image. The permeability of each subvolume was calculated in order to quantify the variation in microstructure within each sample and to determine the minimum size for a representative volume (see Appendix A).

The watershed transformation (Beucher, 1992) is based on the idea of redefining grayscale pixel value as topographic relief. First, interphase boundaries are highlighted by thresholding the grayscale gradient of the denoised image. Then a global threshold is employed to make an initial try at segmenting the denoised data. The image is then inundated starting from the initial segmentation. The regions defined by the thresholded gradient act as impermeable barriers to the rising virtual fluid, preventing the merging of distinctly different phases. The result, after the watershed transformation, is a high-quality, segmented binary image where phase boundaries are defined exactly at grayscale inflections.

The watershed transform is suitable for accurately segmenting larger features in the data; however, it tends to miss very thin melt conduits. To capture these finer details, a top-hat filter (Vincent, 1993) is applied and then a global threshold is utilized to select those details. The size of the kernel is selected based on the size of those features. An opening filter is then applied to the inverse of the image in order to smooth out the boundaries of the image. Some user-controlled refinements of the binary image were typically needed. The size of the features that top-hat segmenting is able to recognize is limited by the kernel size. Avizo limits the size of the kernel to twenty pixels, so a watershed transform is still needed if there are features in the 2-D slice that are larger than the kernel size. Examples of the final 3-D binary images for four charges of different nominal melt fractions are show in Fig. 2.4.

2.3.4 Quantification of network topology

We performed a series of systematic analyses on subvolumes of the 3-D

binary image of our olivine-basalt samples. We quantify the melt fraction, grain size distribution, network interconnectivity, and permeability for each subvolume (Table A.1). The melt fraction (ϕ_m) of each subvolume is measured by calculating fraction of voxels, the three-dimensional image unit, assigned to the melt phase in the segmented image. The measured melt fraction of a subvolume may vary from the nominal melt fraction (ϕ_n) because of sample heterogeneity and possible melt-rock interactions. Uncertainty on the measured melt fraction was estimated by contracting (low bound) and dilating (upper bound) the binary melt image by one pixel (Fusseis et al., 2012). For this reason, error bars are asymmetric.

Grain size distribution was quantified using Avizo's *Separate Objects* module. The module takes the binary label image as input and performs a series of high-level algorithms, including a watershed transform, distance transform, and numerical reconstructions, to separate individual grains by a 1-pixel boundary. We report the grain size distribution for every subvolume as the distribution of equivalent diameters. Separation of individual grains is difficult when melt fraction is low, since the only thing that separates grains are melt channels. Therefore large uncertainties in the equivalent diameter distributions are expected for the scoba-9 ($\phi_n=0.02$) sample.

Quantification of the melt network connectivity was accomplished using Avizo's skeletonization module. Skeletonization is the process by which the general melt microstructure is simplified to an interconnected skeleton network. The skeleton is used to assess the topology of the melt network. First, a distance map is calculated. Second, a thinning algorithm is applied to the binary image that removes pixel-by-pixel the outer layers of melt channels until only a string of pixels remain. The

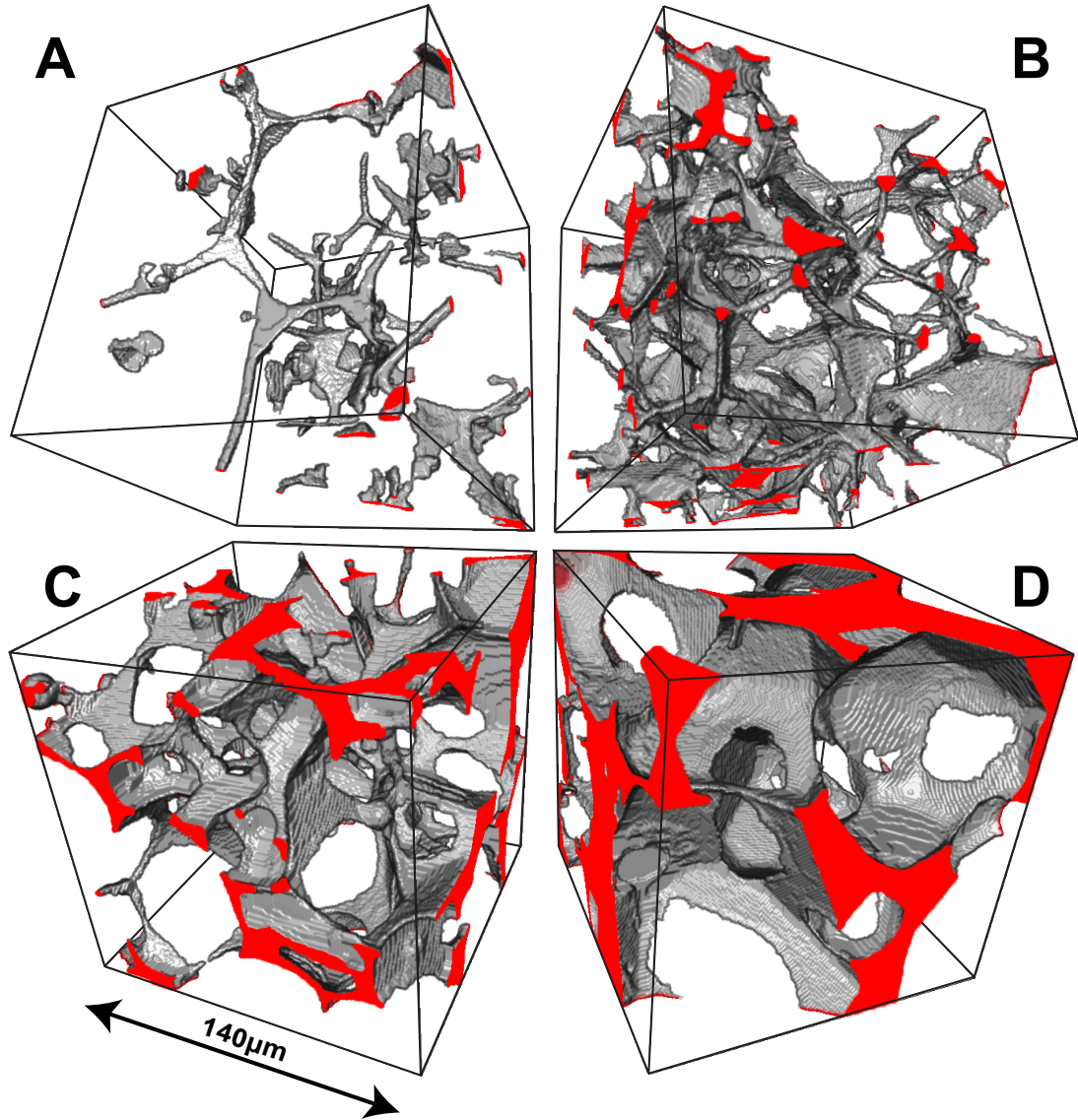


Figure 2.4: Volume renderings of the melt distribution for olivine–basalt containing nominal melt fractions of (A) 0.02, (B) 0.05, (C) 0.10, and (D) 0.20. The dimensions of each subvolume are $140 \times 140 \times 140 \mu\text{m}^3$. Gray represents the melt phase, empty spaces are olivine grains, and red highlights the intersection of melt and the bounding box.

algorithm is calibrated so as to preserve small features of the melt microstructure. Finally, the mean thicknesses of the melt conduits are retrieved from the distance map. A Matlab[®] script, called *ScobaCleaner.m*, was written to automatically remove spurious features from the skeletonized melt network (see supplementary material and Zhu et al. 2011).

2.3.5 Quantification of permeability

Permeability calculations were performed using Avizo's XLab Hydro module. Two different computational modules were used: the Absolute Permeability Experiment Simulation (APES), which computes a scalar estimate of the permeability, and the Absolute Permeability Tensor Calculation (APTC), which computes the 3×3 permeability tensor for the subvolume. Both APES and APTC implement the finite volume method (Harlow and Welch, 1965) to solve the Stokes Equations for the velocity and pressure fields. The Stokes Equations are given by

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \mu \nabla^2 \mathbf{u} - \nabla p = \mathbf{0} \end{cases} \quad (2.2)$$

where p is the pressure [Pa], μ is the viscosity [Pa s] of the simulated fluid, and \mathbf{u} is the velocity [m s^{-1}].

For APES, flow in the digital melt domain is driven by a pressure differential (ΔP) imposed across the subvolume (Fig. 2.5). A 1-pixel-wide impermeable layer is added to the sides of the sample domain parallel to the flow in order to prevent loss of fluid through the adjacent faces. Accommodation zones are added to the inflow and outflow faces of the subvolume to ensure that there is a self-consistent pressure field over the faces. The APES module then solves for the velocity field in the melt domain

(Fig. 2.5). Each APES fluid flow simulation was conducted along the z -direction, parallel to the cylindrical sample axis. During post-processing, the volumetric flux Q [$\text{m}^3 \text{s}^{-1}$] across the sample end faces is computed, and an application of Darcy's Law yields the permeability k [m^2].

$$k = -Q \frac{\mu}{\Delta P} \frac{L}{A} \quad (2.3)$$

where A is the cross-sectional area [m^2] and L [m] is the length of the computational domain.

Contrary to the APES, APTC simulates fluid flow by solving a modified, volume-averaged form of the Stokes Equations (Gray, 1975)

$$\begin{cases} \nabla \cdot \mathbf{D} = \mathbf{0} \\ \mu \nabla^2 \mathbf{D} - \nabla \mathbf{d} = \mathbf{I} \end{cases} \quad (2.4)$$

where \mathbf{D} is a tensorial representation of the spatial deviation of the velocity [s^{-1}], \mathbf{d} is a vectorial representation of the spatial deviation of the pressure [Pa s m^{-1}], and \mathbf{I} is the 3×3 identity matrix. Rather than invoking Darcy's Law, the permeability tensor \mathbf{K} is computed by volume-averaging \mathbf{D} over the whole computational domain V .

$$\mathbf{K} = \frac{1}{V} \int_V \mathbf{D} dV \quad (2.5)$$

Equation systems 2.2 and 2.4 do not lend themselves immediately to solving through implicit methods, since matrices of this form are singular. Therefore, an artificial compressibility coefficient (Chorin, 1967) is incorporated in the discretized forms of Eqn. 2.2 and 2.4.

Differing from the APES module, which imposes a pressure gradient to induce fluid flow, the APTC module supplies mass to the system via a volumetric

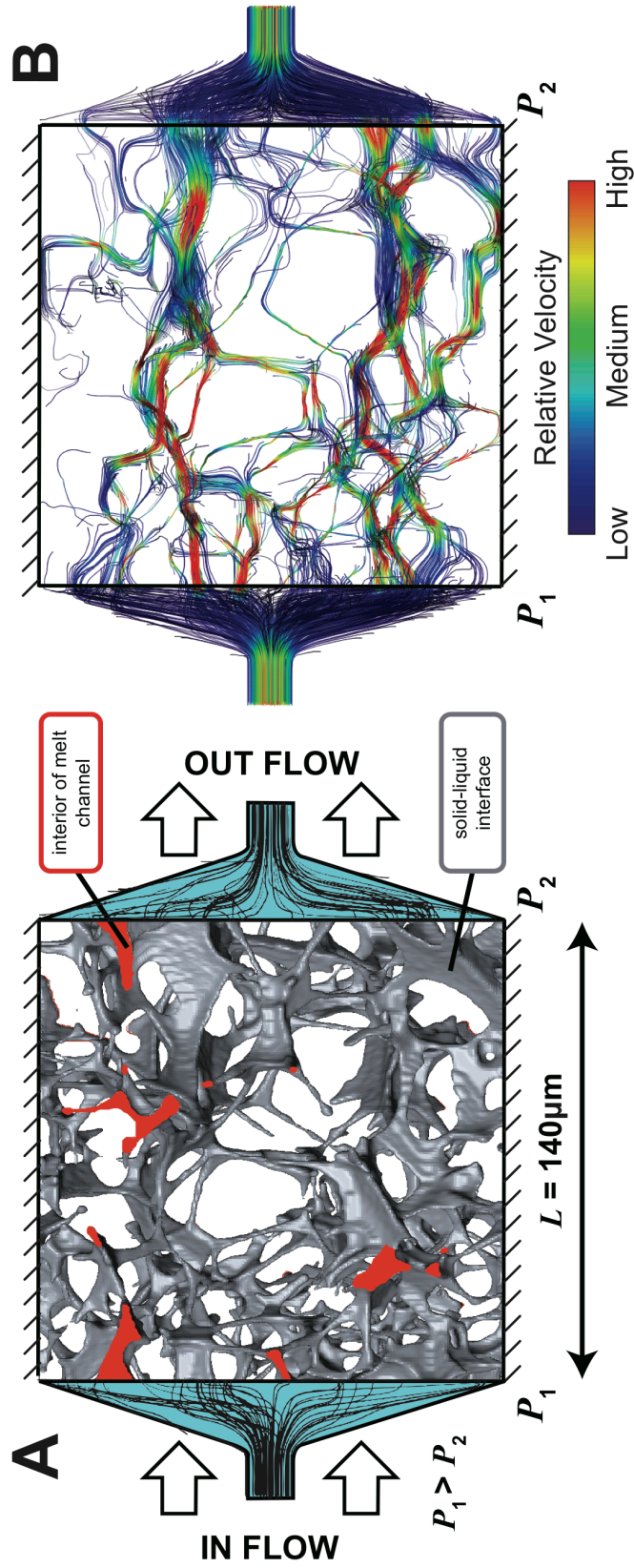


Figure 2.5: (A) Schematic diagram of a virtual permeability experiment conducted using APES module along the z -axis of a $140 \times 140 \times 140 \mu\text{m}^3$ subvolume from scoba-12 ($\phi_a = 0.05$). P is pressure, L is the length of the subvolume, light blue highlights the accommodation zones, and black denotes streamlines. Hash marks indicate impermeable boundaries. (B) The flow field of a virtual fluid is displayed as streamlines and color-coded according to the relative fluid velocity.

source term in the discretized formulation of Eqn. 2.4. Accommodation zones are defined on all six faces of the subvolume to impose periodic boundary conditions between parallel faces. One major drawback of the APTC module is computational cost of the calculation. Moreover, significant permeability anisotropy is not expected in our isostatically pressed samples. The APES module, in contrast, is a relatively quick computation capable of calculating the scalar permeability for a given subvolume, provided the permeability is not significantly anisotropic. For our study, APES is the preferred module for calculating sample permeability. APTC is only used to verify the absence of significant permeability anisotropy.

2.4 Results

The analyses mentioned above were performed on all $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes for nominal melt fractions ranging from 0.02 to 0.20. Refer to the Table A.1 of the online supplement for a complete list of results. From now on, subvolumes will be referred to using the notation “scoba-*a-b-c*”, where the placeholders *a*, *b*, and *c* refer to the sample number, subvolume dimension in pixels, and the subvolume identification number, respectively (Table A.1).

2.4.1 Grain size results

Results from our time series experiments (Appendix A) indicate that the olivine-basalt samples with ϕ_n of 0.02, 0.05, 0.10, and 0.20 have equilibrium textures.

The olivine-basalt aggregates with ϕ_n from 0.05 to 0.2 exhibit lognormal Equivalent Diameter Distributions (EDD). However, the scoba-9 sample ($\phi_n=0.02$)

has an EDD that differs significantly from the others, which likely results from a failure of the *Separate Objects* module to accurately segment individual grains at small melt fractions. For melt fractions as low as 0.02, many of the melt channels are below the resolution of μ -CT (Zhu et al., 2011). When this is the case, two or more adjacent grains may be misrepresented as a single large grain. This may explain why the mean EDD reported for scoba-9 is much larger than the others, and it may also explain why the EDD exhibits a long tail for equivalent diameters larger than 80 μ m. These larger grains cannot be remnants of the pre-sintered samples, since the maximum grain size of the pre-sintered experimental charge is 10 μ m.

The mean equivalent diameters for scoba-9 ($\phi_n=0.02$), scoba-12 ($\phi_n=0.05$), scoba-6 ($\phi_n=0.10$), and scoba-5 ($\phi_n=0.20$) are 42^{+38}_{-20} μ m, 34^{+18}_{-12} μ m, 38^{+21}_{-13} μ m, and 41^{+24}_{-15} μ m, respectively (Fig. 2.6). Errors are asymmetric because equivalent diameter distributions are characteristically lognormal.

2.4.2 Connectivity of melt network

Results from connectivity analyses are conveyed as Coordination Number Distributions (CND) in Fig. 2.7. The skeletonization analysis replaces melt-filled triple junctions with tubules whose widths vary along their axes. The intersections between melt tubules are designated “nodes.” Connectivity is defined as the number of melt tubules connected to each node. The connectivity of an ideal melt network is predicted to be 4 (von Bargen and Waff, 1986), but it varies in natural systems like our samples (Zhu et al., 2011). We determine the CND of one $350\times350\times350\mu\text{m}^3$ subvolume from each sample.

To describe the CND in a physical context, nodes with a coordination number of 1 represent dead-end melt channels. Nodes with a coordination number of 2 are removed from the skeleton, since two connected melt conduits effectively act as one single conduit. Nodes with a coordination number of 3 are mostly associated with regions where melt pooling or grain boundary wetting is occurring. A node with a coordination number of 4 indicates a four-grain junction. Nodes with a coordination number of 5 or higher are either representative of physical junctions in which more than four grains are present, or artifacts from the *ScobaCleaner.m* algorithm when the connections from short tubules get merged (Table A.2).

The CNDs of scoba-5 ($\phi_n=0.20$), scoba-6 ($\phi_n=0.10$), scoba-12 ($\phi_n=0.05$), and scoba-9 ($\phi_n=0.02$) indicate that the frequency of coordination number 4 nodes decreases as melt fraction increases (Fig. 2.7). This represents a decrease in the number of melt junctions connected to four melt tubules. Conversely, the frequency of coordination number 3 increases over the same range, representing an increase in melt grain boundary wetting. The higher connectivity nodes, e.g. 5-8, have more or less the same frequency across scoba-12, scoba-6, and scoba-5.

Scoba-9 ($\phi_n=0.02$) appears to contradict the progression towards a coordination number 4 dominated melt microstructure, since coordination number 3 nodes represent a clear majority of the nodes in the network. However, many thin melt tubules in scoba-9 appear broken and register as nodes having a connectivity of 1. This artifact is a result of the μ -CT resolution limits. A node having four connected tubules might register as a node that has only three connecting tubules if one of those tubules is removed during cleaning or data processing. This would account for the

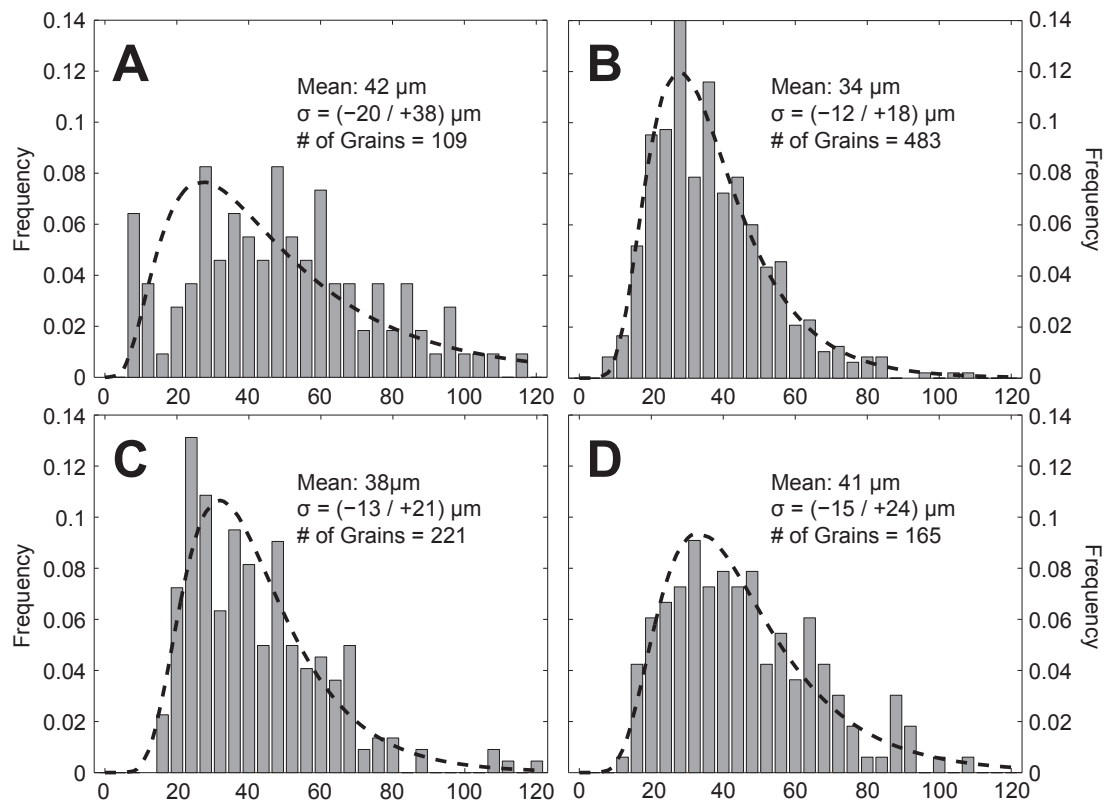


Figure 2.6: Equivalent diameter distributions from $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes of (A) scoba-9-500-4, ($\phi_n = 0.02$), (B) scoba-12-500-1 ($\phi_n = 0.05$), (C) scoba-6-500-4 ($\phi_n = 0.10$), and (D) scoba-5-500-1 ($\phi_n = 0.20$). The geometric mean, geometric standard deviation (σ), and number of grains contained within each sample are reported. Histograms are calculated with 30 bins. Dashed lines represent the best-fit lognormal distributions to the equivalent diameter data.

anomalously high abundance of dead-end tubules as well as the less-than-expected frequency of coordination number 4 nodes. Notwithstanding these resolution limits, it is clear that the melt network remains well connected even when the nominal melt fraction is 0.02 and the measured melt fraction of representative subvolumes approaches $0.0121^{+0.006}_{-0.005}$. Therefore, even at low melt fractions our subvolumes support fluid flow.

2.4.3 Permeability results

Permeability was computed for three to five $350 \times 350 \times 350 \text{ } \mu\text{m}^3$ subvolumes per sample (Fig. 2.3). Fig. 2.8 shows the calculated permeability as a function of the measured melt fraction of each subvolume. We performed a linear fit on the data using the total least squares algorithm based on York et al. (2004), including the standard error on measured melt fraction. Since permeability values were calculation results, no uncertainty was reported. Uncertainty of melt fractions came from the ambiguity in the location of the olivine-basalt phase interface. The upper and lower bounds of melt fractions were estimated by expanding and shrinking the melt phase by 1 pixel at the olivine-melt interface (Fusseis et al., 2012). When fitting the data, we shift the porosity value to halfway between the upper and lower bounds of melt fraction. We find that fluid flow in our olivine-basalt samples is well characterized by a power-law relationship between permeability and melt fraction (Eqn. 2.1), where the power law exponent is $n = 2.6 \pm 0.2(1\sigma)$, and, assuming a grain size of $35 \text{ } \mu\text{m}$ in our samples, the geometric constant is $C = 58^{+36}_{-22} (1\sigma)$.

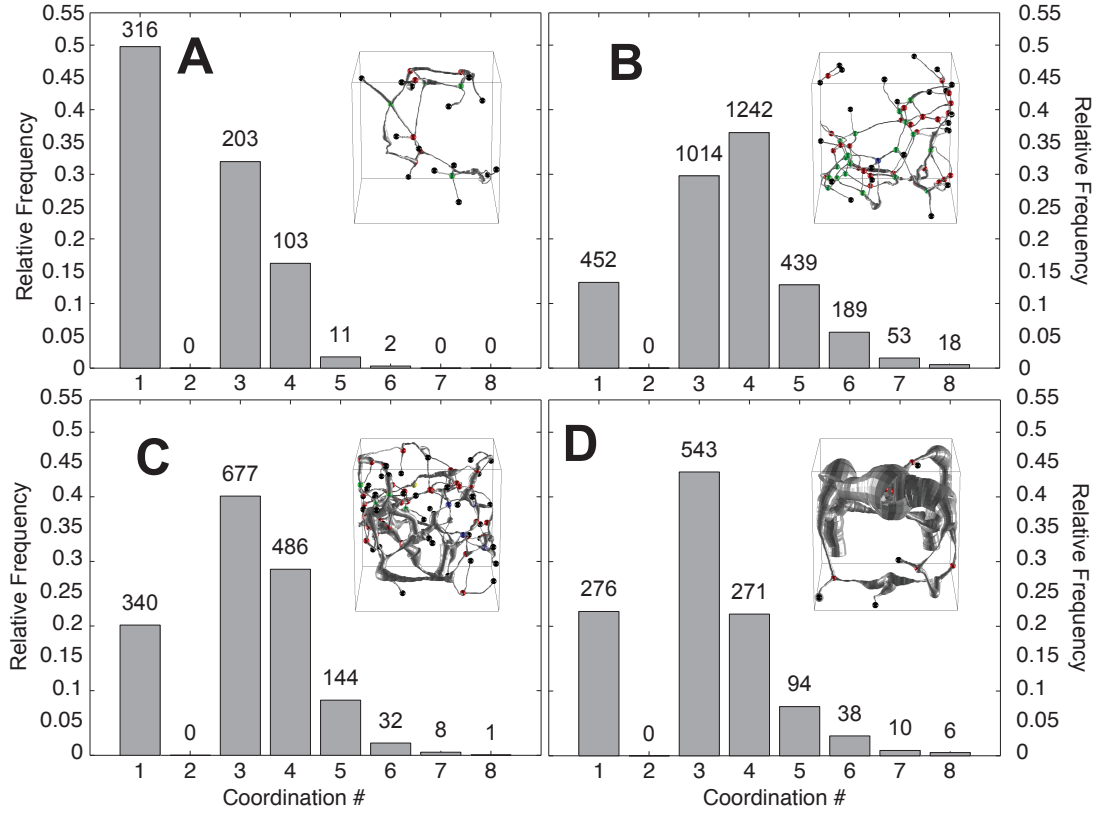


Figure 2.7: Coordination number distribution of melt fraction series experiments. Relative frequency of coordination numbers of 350×350×350 μm³ subvolumes of (A) scoba- 9-500-4 ($\phi_n = 0.02$), (B) scoba-12-500-1 ($\phi_n = 0.05$), (C) scoba-6-500-4 ($\phi_n = 0.10$), and (D) scoba-5-500-1 ($\phi_n = 0.20$). Total counts of each coordination number are reported above each bar. The network skeleton of a representative 105×105×105 μm³ is included as an insert for each sample. The nodes in the skeleton are color-coded according to their coordination number, e.g. 1–black, 3–red, 4–green, 5–blue, 6–magenta, and > 6 – yellow. The radius of the melt tubules in the skeleton visualized in the inserts are proportional to melt conduit thickness in the original, pre-skeletonized melt microstructure.

2.4.4 Permeability Anisotropy

We computed the permeability tensor \mathbf{K} for the scoba-12-500-4 subvolume ($\phi_n=0.05$) using the APTC module, yielding

$$\mathbf{K} = \begin{bmatrix} 1.86 & 2 \times 10^{-3} & -6 \times 10^{-2} \\ 2 \times 10^{-3} & 1.90 & 8 \times 10^{-2} \\ -6 \times 10^{-2} & 8 \times 10^{-2} & 1.94 \end{bmatrix} \times 10^{-15} \text{ m}^2 \quad (2.6)$$

The eigenvalues of \mathbf{K} , called the principal permeabilities, are $2.02 \times 10^{-15} \text{ m}^2$, $1.88 \times 10^{-15} \text{ m}^2$, and $1.81 \times 10^{-15} \text{ m}^2$. The coefficient of variation of these values is $\sim 6\%$, which is negligible compared to modeling uncertainty. Therefore, we conclude that the melt microstructure of our sample is isotropic at the scale of this $350 \times 350 \times 350 \mu\text{m}^3$ subvolume. Since the microstructures are isotropic, we conclude that isostatically pressing the samples produces an isotropic permeability structure, so the APES module is sufficient for computing the permeabilities of our subvolumes.

The permeability of this subvolume determined by the APES module is $4.6 \times 10^{-15} \text{ m}^2$, which is about a factor of 2 larger than the determination from APTC. The discrepancy is likely due to the different formulation of the permeability determination problem. The formulation used by APES is closest to the original definition of permeability and is therefore preferred here. We also artificially rotated the subvolume and recalculated the permeability by APES in three mutually perpendicular directions. We find the permeabilities to be $5.4 \times 10^{-15} \text{ m}^2$, $4.7 \times 10^{-15} \text{ m}^2$, and $4.6 \times 10^{-15} \text{ m}^2$ for k_x , k_y , and k_z , respectively. Permeability values are similar within $\sim 3.9\%$ relative variance, which confirms that the permeability in our samples is essentially isotropic.

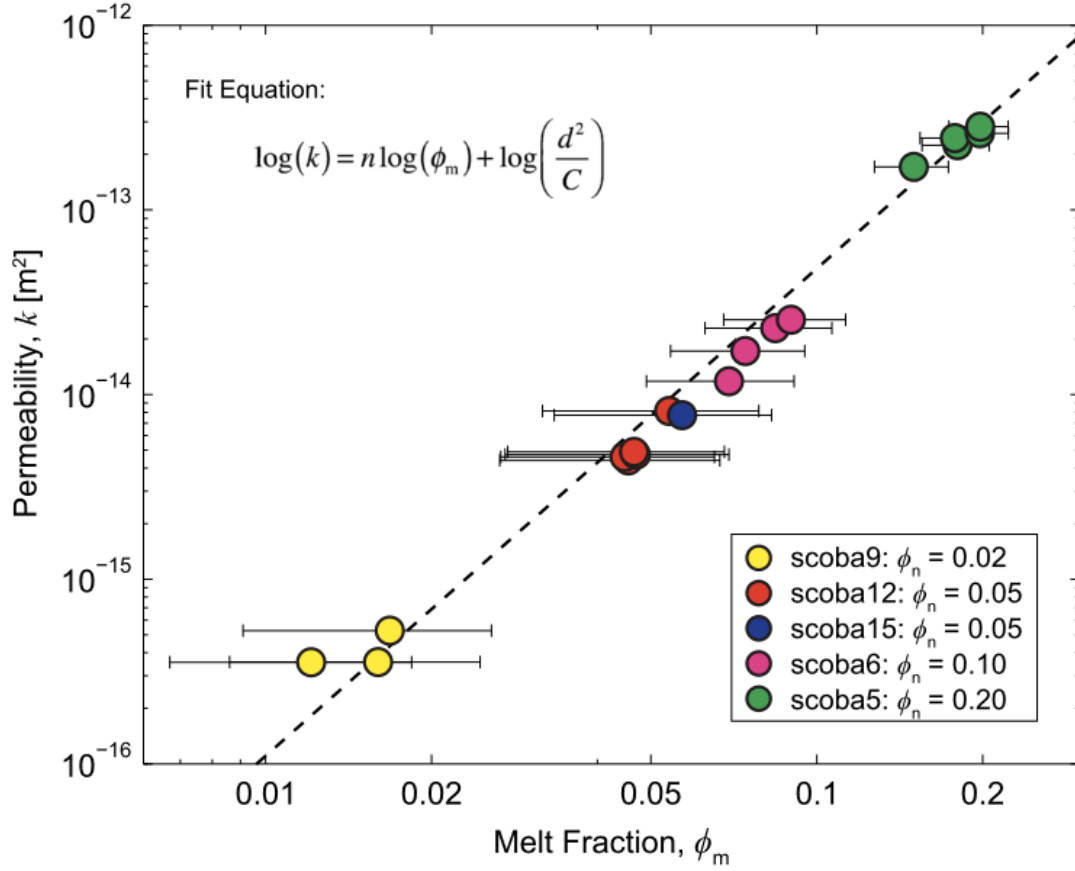


Figure 2.8: Permeability calculated for $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes plotted as a function of the measured melt fraction on log-log axes. Different samples are represented by different colors, with sample number and nominal melt content indicated in the legend. The dashed line represents the best-fit line for $\log_{10}(k) = n \log_{10}(\phi_m) + \log_{10}(d^2/C)$, where geometric constant $C = 58^{+36}_{-22}$ and power law exponent $n = 2.6 \pm 0.2$. For fit, d is assumed to be $35 \mu\text{m}$, a value we chose because it is within the range of grain sizes measured from all subvolumes.

2.5 Discussion

2.5.1 Morphology of melt microstructure

Visual inspection of our melt microstructures reveals that, for low melt fractions the network topology resembles the ideal model proposed by von Bargen and Waff (1986), where melt preferentially reside to three and four-grain junctions. As the nominal melt fraction increases to 0.05, we visually observe the onset of grain boundary wetting, though melt tubules continue to be the dominant feature of the melt microstructure. At $\phi_n=0.10$, there is an inversion from a tube-dominated network to one in which the melt films and pools are the most prominent features. Finally, for $\phi_n=0.20$, grain boundaries are almost completely wetted, as the sample is approaching its theoretical disaggregation limit, $\phi_n \geq 0.20$ (Hier-Majumder et al., 2006; McKenzie, 1984).

2.5.2 Interpretation of power law exponent

The permeability of an ideal melt network, in which grain size is uniform, depends on the square of melt fraction, i.e., $n=2$ when melt resides at triple junction (von Bargen and Waff, 1986) and on the cube of melt fraction, i.e. $n=3$, as higher melt fraction (Wark et al., 2003). This transition may correspond the two morphological regimes observed here, i.e. a tubule-dominated at low melt fractions ($n=2$) versus pool and film-dominated at higher melt fractions ($n=3$). Considerations of grain-scale heterogeneity would also produce $n=3$ (Zhu and Hirth, 2003). However, the data from this study are captured adequately by a single relation with $n = 2.6 \pm 0.2$ and $C = 58^{+36}_{-22}$. More complex relations are not justified by the data,

considering the uncertainty of our porosity and permeability estimates.

The experimental results of Renner et al. (2003) and Connolly et al. (2009) are compatible with $n=3$, which, considering that these experiments infer permeability indirectly from the compaction rate of olivine-basalts aggregate, present an encouraging agreement with our study. Therefore, microstructure readjustment during quenching appears to be minor in our experiments and our permeability–porosity relation can probably be used to describe olivine-basalt aggregates at mantle conditions. For extrapolation to higher temperatures and pressures, we may need to consider an increased importance of melt film grain faces, as the dihedral angle appears to decrease as temperature and pressure increase (Yoshino et al., 2009b). However, melt films observed at high melt fraction in our sample do not have a marked effect on our permeability–porosity relation. Future work would need to address their contribution to permeability at low melt fraction, high pressure, and high temperature.

Given the various melt geometries present in our datasets, a value of $n=2.6$, between 2 and 3, is not surprising. Consider a mixture of subvolumes consisting of end member melt distributions, one end member is entirely made up of melt tubules along triple junctions ($n=2$) while the other contains only wet grain boundaries and melt pools ($n=3$). The overall permeability of the system is the mixing of the individual subvolume permeabilities and, in the absence of a large-scale order between these subvolumes, will converge to the geometric mean permeability as the number of subvolumes increases (Madden, 1976). If the permeability of each subvolume V_i is given by the empirical relation $k_i = C_i \phi^{n_i}$, the geometric mixing leads

to an equation for the total permeability k_T

$$k_T = (C_1 \dots C_N)^{\frac{1}{N}} \phi^{\frac{n_1 + \dots + n_N}{N}} \quad (2.7)$$

where N is the total number of subvolumes. Eqn. 2.7 is, in its own right, a power law, the same as Eqn. 2.1. In our case, our end member distributions have $n=2$ and $n=3$, so Eqn. 2.7 leads to a new power law exponent of 2.5, which is very much consistent with the value of $n = 2.6 \pm 0.2$ obtained by our fit. A value of n between 2 and 3 can be thought of as representing a mixing of melt geometries.

2.5.3 1-D mantle model

Given the new empirical relation between permeability and the melt fraction, we make a simple model of melt transport in the mantle. If ^{230}Th disequilibrium observed is produced at 60 to 75 km depths, melt transport must have occurred at a velocity w of order of 1 m yr^{-1} ($3 \times 10^{-8} \text{ m/s}$). Darcy's law implies

$$\phi w = \frac{k_0}{\mu} \phi^n \Delta \rho g \quad (2.8)$$

where $k_0 = d^2/C$ is the permeability coefficient, ϕ is the porosity, $\Delta \rho$ is the density contrast between melt and solid mantle, $\mu = 10 \text{ Pa s}$ is the melt viscosity (Ryan and Blevins, 1987) and $g \sim 10 \text{ m/s}^2$ is the acceleration of gravity.

Assuming a grain size of 3 mm (Toramaru and Fujii, 1986), we estimate $k_0 \sim 1.55 \times 10^{-7} \text{ m}^2$. If $\rho_s = 3300 \text{ kg m}^{-3}$, $\rho_f = 2700 \text{ kg m}^{-3}$ (Stolper et al., 1981), $\Delta \rho = 600 \text{ kg m}^{-3}$. From Eqn. 2.8, the porosity needed to sustain a melt velocity w is given by

$$\phi = (w/\beta)^{\frac{1}{n-1}} \quad (2.9)$$

where $\beta = k_0 \Delta \rho g / \mu = 9.3 \times 10^{-5} \text{ m s}^{-1}$. Therefore, preserving ^{230}Th disequilibrium

produced at depth requires a porosity of at least 0.0068. This number is comparable with estimates from seismic studies (The MELT Seismic Team, 1998), although at the low end of the observational constraints. Higher porosity results in faster melt velocity, which is more easily reconciled with ^{230}Th excess in MORB.

An alternative estimate of mantle porosity can be obtained from a mass balance between melt produced by decompression of a mantle column at velocity W (Ribe, 1985; Spiegelman and Elliott, 1993):

$$\rho_f \phi w = \rho_s F W \quad (2.10)$$

where F is degree of melting, which increases with height above the level where melt starts. By combining Eqns. 2.8 and 2.10, the mean melt fraction retained by our model mantle is estimated at.

$$\phi = \left(\frac{\rho_s}{\rho_f} \frac{F W}{\beta} \right)^{\frac{1}{n}} \quad (2.11)$$

Remarkably, the permeability in this model does not depend on porosity but only on geodynamical parameters

$$k = \frac{\rho_s}{\rho_f} \frac{\mu F W}{\Delta \rho g} \quad (2.12)$$

Assuming $F=0.20$ at the top of the melting column (Asimow et al., 1995) $W=5$ cm yr $^{-1}$ (1.7×10^{-9} m s $^{-1}$) (Spiegelman and Elliott, 1993), we obtain a melt fraction $\phi=0.0085$, and, according to Eqn. 2.8, a melt velocity of 5.0×10^{-8} m s $^{-1}$ (~ 1.6 m yr $^{-1}$).

If this velocity were valid for the entire melting column, the transit time through the melting column z_M would be

$$t_U = \frac{z_M}{w} = \left(\frac{\rho_s}{\rho_f} F_M W \right)^{\frac{1-n}{n}} \beta^{-1/n} z_M \quad (2.13)$$

where F_M is the degree of melting in the column. However, the degree of melting increases upward in the column. Assuming a linear increase of F from 0 to F_M through a column of height z_M , we obtain

$$t_T = nt_U \quad (2.14)$$

For $F_M=0.2$ and $z_M=75$ km, $t_T \sim 136$ kyrs. This value is in the high end of what is permissible to preserve ^{230}Th excesses, especially considering that chromatographic effect will reduce the velocity of isotopes (Spiegelman and Elliott, 1993). However, the transit time depends on grain size to the power $-2/n$ through the β coefficient. Increasing the grain size to 1 cm reduces the melt transit time to 54 kyrs, although a melt fraction of 0.0034 which is harder to reconcile with geophysical estimates of melt content underneath mid-ocean ridges.

A larger melt fraction would be compatible with ^{230}Th constraints but could not be sustained by melting of an upwelling mantle column. However, these calculations assume a very simple system, i.e. 1-D melt percolation through a uniform network in steady state. They do not give any consideration heterogeneities in the melt distribution larger than the grain-scale. It may be possible to reconcile uranium-series disequilibrium and geophysical observations if the mantle is heterogeneous, with high porosity channels.

2.5.4 Implications for mantle heterogeneities

High melt fraction dunite conduits have been observed in ophiolites and

appear necessary to explain chemical disequilibrium between mid-ocean ridge basalts and the mantle residuum (Dick, 1977; Johnson and Dick, 1992; Kelemen et al., 1992; Quick, 1982; Spiegelman and Kelemen, 2003). Dunite conduits form as a buoyant melt, which is saturated in olivine but under-saturated in orthopyroxene (Ortoleva et al., 1987), reacts with pyroxene-bearing peridotite, simultaneously dissolving the orthopyroxene and precipitating olivine (Kelemen et al., 1997, 1995a, 1995b). The dissolution of pyroxene is an incongruent melting reaction: more melt is produced by volume than is removed from the system by the precipitation of olivine (Kelemen et al., 1995b), and increases both melt fraction and permeability. Naturally, the rate of dissolution is enhanced in regions where permeability is increased, which, in turn continues to enhance permeability. Thus, a positive feedback, known as the reactive infiltration instability (RII), is established between the opx dissolution and permeability enhancement. Numerical models (Aharonov et al., 1995; Kelemen et al., 1997; Spiegelman et al., 2001; Spiegelman and Kelemen, 2003) have shown that the RII is capable of producing banded dunite structures similar to those found in nature.

Our results have direct implications for melt transport within these conduits. At the grain-scale, permeability is largely controlled by the local melt distribution, which is determined by local variations in the free surface energy of the system. Free surface energy is an intrinsic property of the system composition, i.e. the mineral phases present and the composition of the melt. Since the compositions of our samples are similar to those of partially molten dunite, it stands to reason that melt transport within these dunite conduits adheres to the power-law relationship between permeability and melt fraction that we constrain here. Due to the RII, the melt

fraction within dunite conduits is four times the overall mantle melt content (Spiegelman et al., 2001). Therefore, the permeability of these conduits is about 37 times larger than for a homogeneous mantle. Neglecting melt production by RII, the channels would occupy 25% of the mantle, so that channelization would increase the velocity by approximately a factor of 10, making it easier to preserve ^{230}Th disequilibrium while verifying the mass balance considerations described in the previous section.

The permeability of dunite conduits may further increase if the difference in surface energy between olivine and opx is sufficient to preferentially partition melt to olivine-rich areas (Watson, 1999), increasing melt content in dunite conduit beyond the product of incongruent melting. Lithological melt partitioning has been proposed to occur in mantle systems where olivine and opx are present. However, experimental evidence for melt partitioning in systems with mineralogies similar to the mantle is lacking. Although more research is needed to establish the extent to which the RII and lithological partitioning modify the permeability structure of the mantle, dunite conduits are good candidates for enhancing overall melt transport within the partially molten region of the mantle beneath mid-ocean ridges.

2.6 Conclusion

This study is the first to use a 3-D imaging technique on synthetic partially molten peridotites to estimate sample permeability. Visual inspection of the digital microstructures shows that for melt fractions as low as 0.02, interconnected melt channels residing along grain edges are the dominant features of the melt network.

For melt fractions greater than 0.05, considerable melt pooling and grain boundary wetting are observed in addition to melt channels. Measured connectivity distributions confirm the increased contribution of grain boundary wetting as melt content increases.

The permeability of our samples was computed numerically for sufficiently large representative subvolumes and ranged from 4×10^{-16} to $2 \times 10^{-13} \text{ m}^2$ for melt fractions ranging from 0.02 to 0.20. The relationship between permeability and local melt fraction is adequately represented by a power law $k = d^2 \phi^n / C$, with d the grain size (approximately $35 \text{ }\mu\text{m}$ in our samples), the exponent $n = 2.6 \pm 0.2$, and the geometric constant $C = 58^{+36}_{-22}$. A first-order calculation, based on mass balance in a 1-D melting column, show that our empirical relation implies a melt fractions of order 1% under mid-ocean ridges with upwelling velocities of order 1 m yr^{-1} leading to transit times through the melting column that are consistent with those constrained by uranium-series analyses. Combined with numerical computation, μ -CT has proven to be a useful tool for characterizing the microstructure of partially molten peridotites and computing their material properties. The results of this study place important new constraints on melt transport beneath mid-ocean ridges, where partial melting occurs.

Chapter 3: Influence of microstructure on electrical conductivity of partially molten rocks

Abstract

Estimates of melt content beneath fast-spreading mid-ocean ridges inferred from magnetotelluric tomography (MT) vary between 0.01 and 0.10. Much of this variation may stem from a lack of understanding of how the grain-scale melt geometry influences the bulk electrical conductivity of a partially molten rock, especially at low melt fraction. We compute bulk electrical conductivity of olivine-basalt aggregates over 0.02 to 0.20 melt fraction by simulating electric current in experimentally obtained partially molten geometries. Olivine-basalt aggregates were synthesized by hot-pressing San Carlos olivine and high-alumina basalt in a solid-medium piston-cylinder apparatus. Run conditions for experimental charges were 1.5 GPa and 1350 °C. Upon completion, charges were quenched and cored. Samples were imaged using synchrotron X-ray micro-computed tomography (μ -CT). The resulting high-resolution three-dimensional (3-D) image of the melt distribution constitutes a digital rock sample, on which numerical simulations can be conducted to estimate material properties. To compute bulk electrical conductivity, we simulated a direct current measurement by solving the current continuity equation, assuming electrical conductivities for olivine and melt. An application of Ohm's Law yields the bulk electrical conductivity of the partially molten region. The bulk electrical conductivity values for nominally dry materials follow a power-law relationship $\sigma_{\text{bulk}} = A\sigma_{\text{melt}}\phi^m$ with fit parameters $m = 1.3 \pm 0.3$ and $A = 0.66 \pm 0.06$. Laminar fluid flow

simulations were conducted on the same partially molten geometries to obtain permeability, and the respective pathways for electrical current and fluid flow over the same melt geometry were compared. Our results indicate that the pathways for flow fluid can be different from those for electric currents. The tortuosity of direct current pathways is lower than that of fluid flow pathways. The simulation results are compared to existing experimental data, and the potential influence of volatiles and melt films on electrical conductivity of partially molten rocks are discussed.

3.1 Introduction

At mid-ocean ridges, melt is thought to percolate over a broad, partially molten region through a grain-scale network of interconnected melt (Fig. 3.1). The capacity of the upper mantle to transport melt, which is ultimately responsible for the production of oceanic crust, strongly depends on the spatial distribution of melt in the mantle. The magnetotelluric (MT) method, which exploits the high conductivity of partially molten rock, is a valuable tool used to probe the melt content of the upper mantle. Though MT measurements are consistent with the presence of partial melt at mid-ocean ridges, they disagree on the shape of the melting region and on the local melt fraction, with estimates in the literature varying from as low as 0.01-0.03 (Evans et al., 1999) to as much as 0.10 (Key et al., 2013). The first step towards reconciling MT survey estimates is to robustly link electrical conductivity of partially molten mantle rocks to the grain-scale morphology and interconnectivity of melt. A microstructure-based approach to constraining electrical conductivity as a function of melt fraction will provide a baseline for extrapolate laboratory measurement to

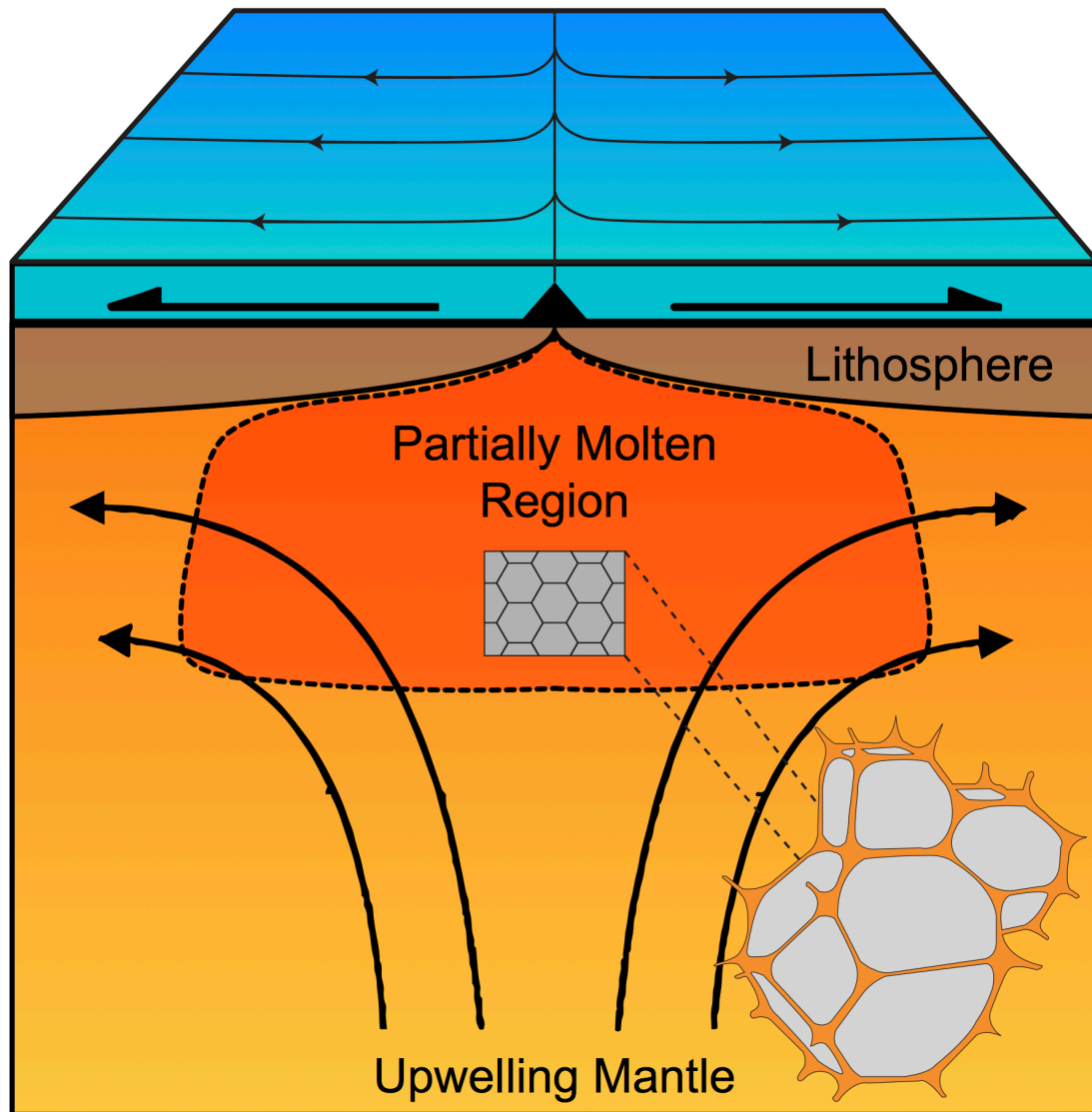


Figure 3.1: Schematic diagram of a symmetrically spreading mid-ocean ridge. Surface arrows denote the spreading direction, curved arrows denote the upwelling direction of the mantle, blue region represents the ocean, brown represents the lithosphere, orange gradient represents the asthenosphere, and red represents the partially molten region beneath the ridge axis. Modified from Weatherley, (2012). Pop-out figure is a depiction of an idealized, grain-scale melt geometry that is modified from Toramaru and Fujii (1986).

natural conditions and to assess the potential contributions of volatiles and melt anisotropy to bulk electrical conductivity.

Bulk electrical conductivity of partially molten rock strongly depends on interconnectivity of the highly conductive melt phase. For a monomineralic system, under hydrostatic melting conditions, melt settles into an equilibrium configuration that minimizes the total surface energy of the system. The degree of interconnectivity can be assessed by the dihedral angle associated with its constituent solid-liquid phase boundaries (Bulau et al., 1979; Waff and Bulau, 1979). For a dihedral angle greater than 60° , melt forms isolated pockets. In this case, the melt and solid phases are connected in series and the bulk electrical conductivity of the mixture is only marginally greater than that of the solid. However, for a dihedral angle less than 60° , as is the case for a partially molten olivine-basalt (Cmíral et al., 1998; Cooper and Kohlstedt, 1984; Jurewicz and Jurewicz, 1986; Toramaru and Fujii, 1986; Waff and Bulau, 1982), melt forms an interconnected network along grain edges (von Bagen and Waff, 1986). As such, the melt conducts electricity in parallel with olivine and the bulk electrical conductivity for melt fractions greater than 0.01 increases by at least one order of magnitude (Roberts and Tyburczy, 1999; ten Grotenhuis et al., 2005; Yoshino et al., 2010).

Since the electrical conductivity of rock strongly depends on the melt geometry, bulk conductivity versus melt fraction relationships have been derived for a number of idealized melt geometries: cube pack (Waff, 1974), tube lattice (Schmeling, 1986), and hard sphere pack (Hashin and Shtrikman, 1963; Waff, 1974). Though these end-member cases are useful for conceptualizing melt configurations,

partially molten mantle rocks are heterogeneous and exhibit a range of melt features (Faul, 2000; Laporte and Provost, 2000; Miller et al., 2014; Wark et al., 2003) depending on the melt fraction present. At melt fraction larger than ~ 0.01 , melt mostly resides in triple junctions connected at four-grain junctions (Miller et al., 2014; Toramaru and Fujii, 1986; Waff and Bulau, 1982, 1979; Zhu et al., 2011) though melt films that wet two-grain boundaries have also been observed at low melt fraction (Cmíral et al., 1998; Faul, 2000; Garapić et al., 2013). Melt pools exist with increasing frequency as melt fraction increases, leading to an increased degree of grain boundary wetting or spillover from triple junctions (e.g., Zhu et al., 2011). At melt fraction of 0.2, melt pools are the dominant feature of the melt network (Miller et al., 2014; Zhu et al., 2011). The coexistence of multiple geometries for a given melt fraction highlights the importance to consider realistic, three-dimensional (3-D) melt geometries when computing material properties like electrical conductivity.

Experiments conducted on partially molten olivine-basalts find that bulk electrical conductivity varies as a power law with melt fraction (i.e., Archie's Law):

$$\sigma_{\text{bulk}} = A \sigma_{\text{melt}} \phi^m \quad (3.1)$$

where σ_{bulk} is bulk conductivity, σ_{melt} is melt conductivity, and ϕ is melt fraction. A and m are power law parameters that depend on the melt morphology and interconnectivity. Values $m=0.89$ to 1.30 and $A=0.73$ to 1.47 have been reported for olivine-basalt partial melts (Roberts and Tyburczy, 1999; ten Grotenhuis et al., 2005; Yoshino et al., 2010). These studies do not directly link electrical conductivity with the melt network morphology.

Most studies find that the data on partially molten samples overlap the upper

Hashin-Shtrikman bound linking the conductivities of pure olivine and melt end-members. However, the upper Hashin-Shtrikman bound is intended to represent a loose pack of uniformly wetted spheres. We argue that this interpretation is inconsistent with microstructural observations of texturally equilibrated rocks (e.g. Cmíral et al., 1998; Cooper and Kohlstedt, 1984; Jurewicz and Jurewicz, 1986; Toramaru and Fujii, 1986; Waff and Bulau, 1982). Also, end-member conductivities were not always directly measured as part of the experiments. While experimental constraints on the electrical conductivity of partially molten rock as a function of melt fraction are essential to interpret MT data, a direct link between electrical properties and melt geometry is still missing.

In addition, the use of electrical conductivity to infer permeability of systems where direct permeability measurements could be challenging, such as partially molten rocks, has garnered significant interest. With the assumptions that pathways for both conductivity and permeability are linked to the microstructure of the rock, several studies have discussed the apparent formation factor, defined as the $\sigma_{\text{bulk}}/\sigma_{\text{melt}}$ and its relation to microstructure in various porous media (Avellaneda and Torquato, 1991; Johnson et al., 1986; Katz and Thompson, 1987). A self-consistent analysis of permeability and electrical conductivity using network (David, 1993) and laminar flow models on periodic pore spaces (Martys and Garboczi, 1992; Schwartz et al., 1993) conclude that these approaches produce comparable results in terms of extrapolating permeability from electrical conductivity. However, considering the fundamental differences in the physics of electrical conduction and fluid flow, it is important to examine the link between electrical conductivity and permeability based

on microstructure.

In this study, we compute the bulk electrical conductivity and permeability of digital rocks that represent the real 3-D distribution of melt in olivine-basalt samples synthesized at mantle pressure-temperature conditions. Each sample was digitized by high-resolution, 3-D imaging using synchrotron-based X-ray micro-computed tomography (μ -CT) (Zhu et al., 2011). The resulting 3-D images constitute digital rocks, on which direct current and fluid flow simulations were conducted. The potential influence of melt films at two-grain boundaries, which have been observed with high-resolution microscopy, on electrical conductivity and permeability is evaluated. We separately assess the influence of H₂O in melt and olivine by adjusting the electrical conductivity of olivine and melt.

3.2 Methods

3.2.1 Sample preparation and imaging

The samples considered in this study are synthetic olivine-basalts aggregates representing partially molten rocks (Miller et al., 2014; Zhu et al., 2011). Experimental charges were prepared from a powdered mixture of San Carlos olivine and natural, Fo₉₀, high-alumina basalt (Mg # = 0.0705) mixed in proportion to achieve nominal melt fractions 0.02, 0.05, 0.10, and 0.20. Charges were isostatically hot-pressed under simulated mantle pressure-temperature conditions (1.5 GPa and 1350 °C) in a solid-medium piston-cylinder apparatus for a minimum of 1 week to achieve textural equilibrium. Upon completion, charges were quenched, turning the molten basalt to glass, and ~1 mm cores were drilled from the samples. Cores were imaged

using a combination of absorption-contrast and phase-contrast X-ray μ -CT at 27 keV to resolve the small density contrast between olivine and basaltic glass. Projections of the integrated X-ray absorption and phase shift were recorded over 180° at 0.12° increments and reconstructed into 3-D grayscale datasets using GridRec (Dowd et al., 1999). Voxel (3-D pixel) values in the reconstructed images roughly correspond to material density. Cubic voxels are 700 nm in length, measured along the voxel edge.

3.2.2 Subvolume selection

Sample cores often exhibit significant decompression cracking. These cracks are voids that are not present at elevated pressure and temperature. To circumvent decompression cracks – and to reduce the size of the computational domain – we consider smaller subsets, or subvolumes, that are cropped from the whole-sample images (Miller et al., 2014). All the subvolumes used in direct current simulations, with the exception of those we used to assess the potential influence of H_2O , have dimensions $280\ \mu\text{m} \times 280\ \mu\text{m} \times 280\ \mu\text{m}$, which was determined to be representative of the bulk based on an electrical conductivity convergence analysis conducted on progressively larger, nested subvolumes (Fig. 3.2). At least three statistically representative subvolumes were cropped from each sample.

3.2.3 Noise-removal and segmentation

Grayscale subvolumes were processed using an edge-preserving anisotropic diffusion filter (Weickert et al., 1998) to remove noise and artifacts, improving the

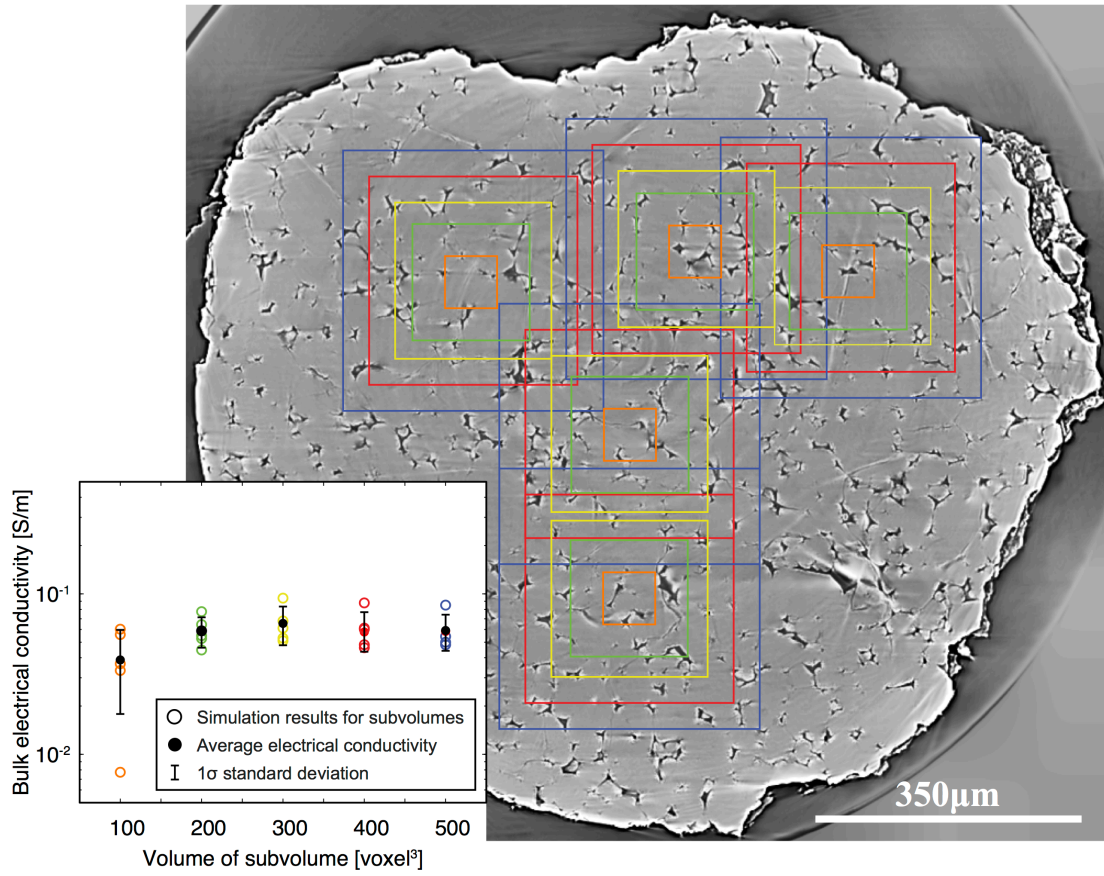


Figure 3.2: Determination of a representative subvolume. Background is a cross-sectional tomography slice of a sample containing melt fraction 0.05. Slice was sampled along the radial-azimuthal plane of the cylindrical sample. Cubic subvolumes of various sizes were cropped from the sample and their bulk conductivities were computed, assuming olivine is a perfect insulator, and plotted as a function of volume (foreground). All subsequent computation are conducted on subvolume 400x400x400 voxel³ (280 μm)³.

efficacy of automatic segmentation algorithms. In order to setup a numerical domain for computation, grayscale subvolume data were transformed into label images using a variety of semi-automatic segmentation techniques: watershed transform (Beucher and Meyer, 1992; Beucher, 1992) for high melt fraction and a bottom-hat global threshold (Vincent, 1993) for low melt fraction. Refer to Miller et al. (2014) for more details in data processing.

The melt fraction of each subvolume was calculated by counting the number of cubic, uniform voxels labeled as basalt. A robust uncertainty analysis of the measured melt fraction requires access to the point-spread function of the image data, which is difficult to obtain. As an alternative, following Füsseis et al. (2012), we estimate lower and upper bounds for the melt fraction by measuring the melt fraction associated with the contracted and dilated melt image, respectively. Contractions and dilations were conducted along all three orthogonal directions of the cubic subvolume.

3.2.4 Direct current simulations

Though the electrical response of a partially molten rock is controlled by the variable mobility of charge carriers to an alternating electric field – either by ambient electromagnetic waves in the Earth or an alternating current source in the laboratory – we chose to simulate direct current only to obtain bulk electrical conductivity. Bulk electrical conductivity should not depend on the type of electrical source, whether it is inferred from the frequency-dependence of alternating current measurements or the direct current simulations. We focus on modeling charge transport by solving the

current continuity equation and do not explicitly consider the mobility of charge carriers.

Our model is based on the formulations proposed by Garboczi (1998) and Zhan et al. (2010). Each segmented label image is considered the computational domain in a direct current simulation. We solve the current continuity equation, which is the Laplace Equation

$$\nabla \cdot (\sigma \nabla \psi) = 0 \quad (3.2)$$

where σ is the local electrical conductivity [S m^{-1}] of voxels associated with each conductive material and ψ is the local scalar electric potential [V] defined at voxel centers. Electric current is driven by an imposed electric potential differential ($\Delta\psi$) across the subvolume, between the inlet and outlet faces. A no-flux condition is imposed at the four faces parallel to the global electric potential gradient to ensure current is conserved (Fig. 3.3). Using a second-order centered finite-difference formulation, Eqn. (3.2) at each voxel becomes

$$\sum_{j=1}^n \kappa_{ij} (\psi_j - \psi_i) = 0 \quad (3.3)$$

where n is the number of connecting voxels and κ_{ij} is the electrical conductance of the bond connecting voxels i and j . The distinction between electrical conductance and electrical conductivity is a geometric factor, which is unity for bonds connecting voxels in a uniform cubic grid. Voxels i and j are restricted to adjoining elements.

With consideration of the no-flux and inlet/outlet conditions, Eqn. (3.3) is reformulated into a matrix equation

$$\kappa_{im} \psi_m = b_m \quad (3.4)$$

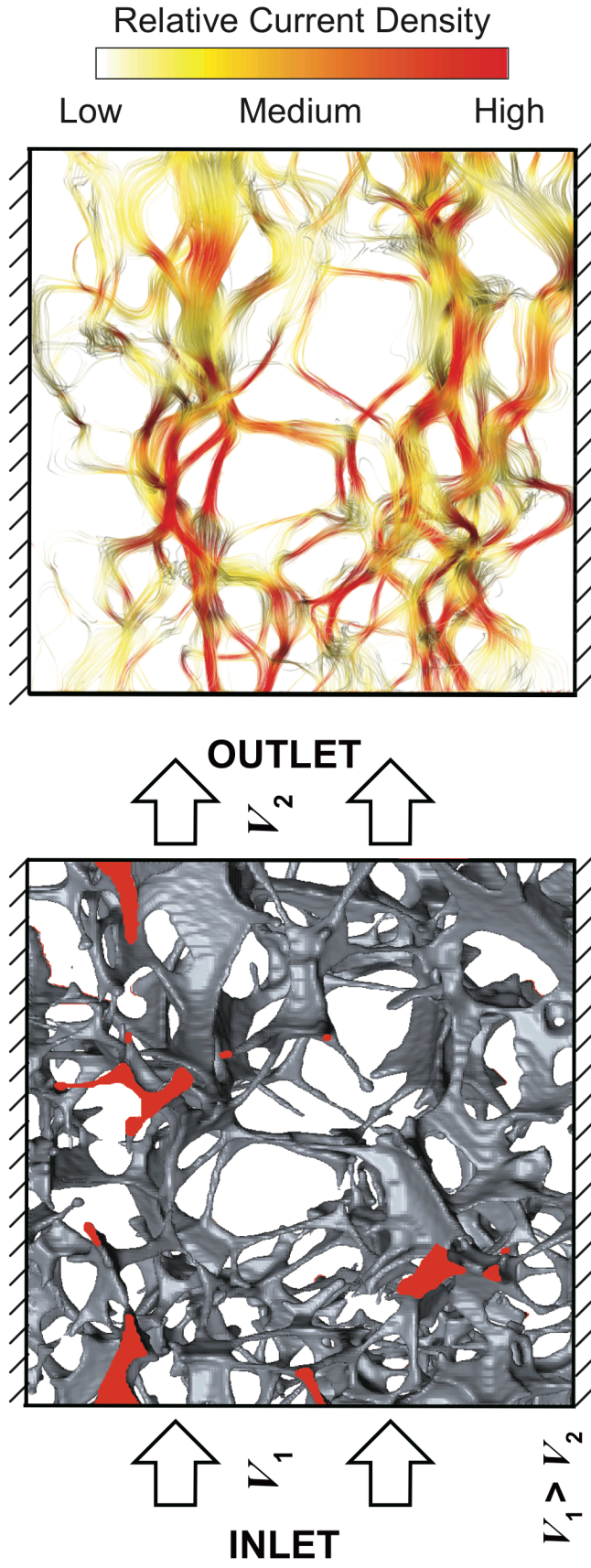


Figure 3.3: Configuration of a numerical direct current experiment conducted along the x-axis of a $140 \times 140 \times 140 \text{ } \mu\text{m}^3$ subvolume that contains $\phi_n = 0.05$. (Left) Grey features are melt. Empty space represents solid grains. Red features denote the intersection of the melt with the bounding box. Arrows represent the flow direction of electric current across the model as a whole, driven by the difference in electric potential V_1 on the inlet side and V_2 on the outlet side. Hash marks illustrate the no-flux condition imposed on the intersections of the bounding box and material interface. (Right) Current density field is displayed as streamlines and color-coded according to the relative magnitude. Absolute magnitude is not important for visualization. In this simulation, olivine is considered a perfect insulator.

where b_m is a vector that contains the influence of the boundary conditions on interior voxels and κ_{lm} is a positive definite, symmetric matrix that contains the electrical conductances of the bonds connecting each voxel. Elements are summed over m indices. An additional constraint on the system comes from current continuity, which states that the conductance of each bond must satisfy

$$\kappa_{lm} = \frac{2\sigma_l\sigma_m}{\sigma_l + \sigma_m} \quad (3.5)$$

If voxels l and m belong to the same material, the conductance of the connecting bond is just the electrical conductivity of that material. Conductance between voxels that are not neighbors equals zero. Eqn. (3.4) is solved using the conjugate gradient method to a tolerance of 1×10^{-5} . An incomplete Cholesky factorization (Meijerink and van der Vorst, 1977) was used as a preconditioner to improve convergence rate of the conjugate gradient solver. Each simulation was set-up, discretized, and solved using custom, Matlab-based finite-difference software.

Evaluating the effect of melt films along grain boundaries requires a special procedure since the resolution of μ -CT is not sufficient to observe possible nanometer scale melt films. We employ an upper bound approach. First we use Avizo's *Separate Objects* module, based on the morphological watershed transform, to define likely olivine-olivine grain boundaries. Assuming all the interfaces are covered by melt films, each voxel at grain boundaries thus consists of both olivine and melt. We assign to these voxels an electrical conductivity that is the parallel average of the olivine and basalt conductivities,

$$\sigma_{\text{film}} = \sigma_{\text{melt}}\chi + \sigma_{\text{olivine}}(1 - \chi) \quad (3.6)$$

where χ is the proportion of the voxel that is occupied by melt. Assuming a melt film

thickness of 100 nm, the maximum value reported in the literature (Cmíral et al., 1998), and considering that our voxels have a uniform thickness of 700 nm, $\chi=1/7$. This approach is similar to that taken by Zhan et al. (2010) to model the effect of an electric double layer on bulk electrical conductivity of sandstone. This approach overestimates the effects of melt films along olivine-olivine grain boundary as the effective conductivity of the voxels should be anisotropic and Eq. 6 should only be valid in the grain-parallel direction. We are able to bracket the effect of the melt films by comparing simulations with $\chi=0$ and $\chi=1/7$.

3.2.5 Fluid flow simulations

Permeability simulations were conducted using Avizo XLab Hydro following Miller et al. (2014). In these simulations, accommodation zones, where fluid spreads evenly over the inlet and outlet faces, were appended to the sample subvolumes. The Stokes Equations, which assume steady-state laminar flow, were solved on a staggered finite-volume grid (Harlow and Welch, 1965). Flow was induced by imposing pressure drop ΔP across the input and output faces. A no-flux condition was imposed at the material interface and the intersection of the melt geometry with the bounding box. As XLab Hydro cannot consider variations in material properties we could not evaluate the effects of melt films using a similar strategy as in the direct current simulations. Instead, we assigned a 1-voxel thick surface along the olivine-olivine boundaries as melt. This approach grossly exaggerates the effect of melt films, which are no more than 1/7 voxel thick. An alternative approach would have been to resample the melt geometry to a voxel size that is comparable to the actual melt film

thickness (tens of nanometers). However, this approach would increase the number of degrees of freedom in our system to an unmanageable size, and as shown later, the exaggerated melt films, as handled by our simplified approach, alter both the permeability and the porosity in the simulations, with negligible effect on the porosity-permeability relationship of partially molten rocks.

3.2.6 Computing tortuosity

Since we solve for the velocity and electrical fields, it is straightforward to compute the tortuosity of each simulation. Tortuosity is defined as the ratio of length of the path a parcel of fluid – or electron for direct current simulations – would travel through the geometry to the length of that geometry in the direction parallel to flow. The tortuosity can be recovered by computing the path length of streamlines, since streamlines are also pathlines for laminar flow. The streamlines can be weighted by its associated mass flux (Matyka et al., 2008). If we take the limit as the spacing between the streamline seeds goes to zero, as would be the case in a continuum, the tortuosity can be calculated using,

$$\tau = \frac{\langle u_{\text{mag}} \rangle}{\langle u_z \rangle} \quad (3.7)$$

where u_{mag} is the velocity magnitude and u_z is the z -component of the velocity, assuming z is the direction of flow (Duda et al., 2011).

3.3 Results

3.3.1 Electrical conductivity

Bulk electrical conductivity was computed for each subvolume label image

Nominal Melt Fraction	Subvolume ID	Measured Melt Fraction, ϕ	σ_{bulk} [S m^{-1}]	Permeability, k [m^2]
0.2	2	0.1501 (-0.0285 / +0.0297)	3.90×10^{-1}	1.45×10^{-13}
0.2	3	0.1980 (-0.0285 / +0.0291)	5.25×10^{-1}	2.48×10^{-13}
0.2	4	0.1960 (-0.0296 / +0.0302)	5.63×10^{-1}	2.85×10^{-13}
0.05	2	0.0539 (-0.0256 / +0.0298)	1.01×10^{-1}	7.00×10^{-15}
0.05	3	0.0438 (-0.0212 / +0.0255)	6.13×10^{-2}	4.02×10^{-15}
0.05	4	0.0439 (-0.0207 / +0.0245)	6.30×10^{-2}	3.69×10^{-15}
0.05	5	0.0465 (-0.0221 / +0.0261)	7.29×10^{-2}	4.25×10^{-15}
0.05	1	0.0562 (-0.0282 / +0.0322)	1.05×10^{-1}	7.29×10^{-15}
0.2	1	0.1824 (-0.0296 / +0.0305)	5.05×10^{-1}	2.47×10^{-13}
0.05	1	0.0485 (-0.0249 / +0.0299)	7.50×10^{-2}	4.70×10^{-15}
0.1	2	0.0855 (-0.0267 / +0.0289)	1.55×10^{-1}	2.15×10^{-14}
0.1	3	0.0845 (-0.0256 / +0.0277)	1.62×10^{-1}	2.32×10^{-14}
0.1	4	0.0770 (-0.0244 / +0.0266)	1.29×10^{-1}	1.56×10^{-14}
0.02	2	0.0164 (-0.009 / +0.0111)	1.47×10^{-2}	3.24×10^{-16}
0.02	3	0.0140 (-0.0077 / +0.0097)	1.35×10^{-2}	N/A
0.02	4	0.0115 (-0.0061 / +0.0076)	1.38×10^{-2}	N/A
0.2	5	0.1986 (-0.0296 / +0.0302)	5.69×10^{-1}	2.66×10^{-13}

Table 3.1: Direct current (using $\sigma_{\text{melt}} = 7.53 \text{ S m}^{-1}$ and $\sigma_{\text{olivine}} = 0.009$) and fluid flow simulations results, including measured melt fractions, bulk electrical conductivity, and permeability.

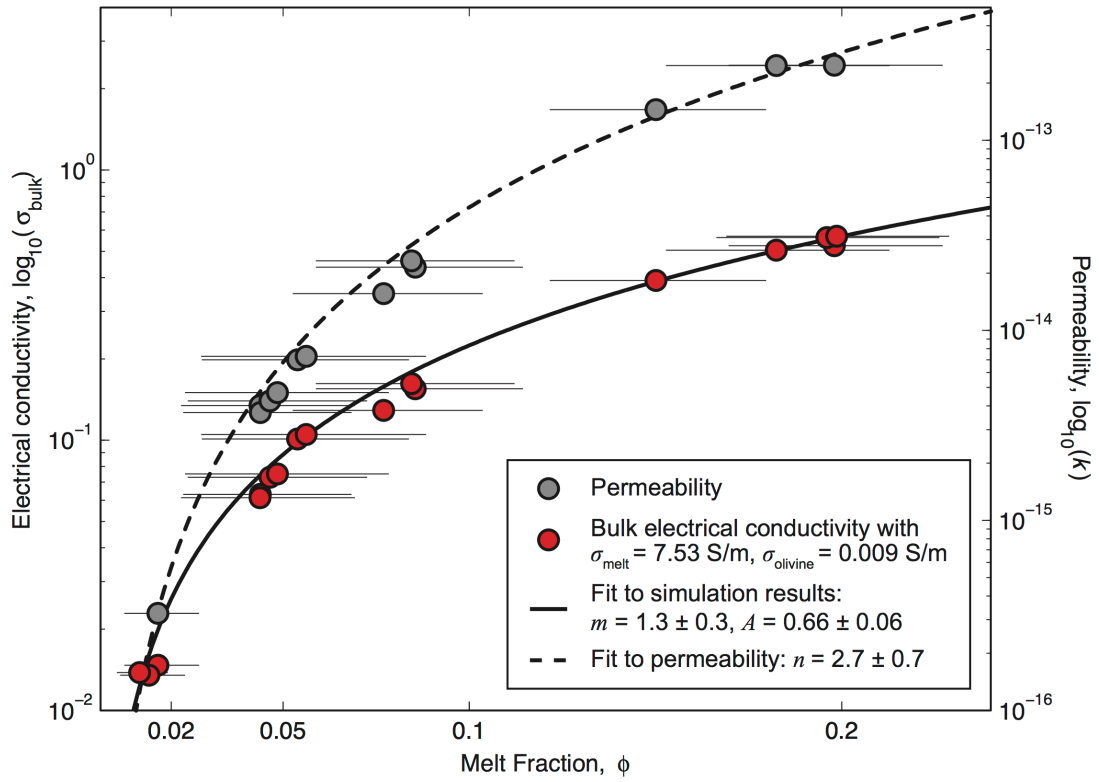


Figure 3.4: Relations between electrical conductivity, permeability, and melt fraction reported on a log-linear scale. Results are fitted to Archie power-law relations. Permeabilities are fitted to power laws $k=C\phi^n$, where C and n are fit parameters.

(280 μm)³, assuming conductivities of 7.53 S m⁻¹ (ten Grotenhuis et al., 2005) and 0.009 S m⁻¹ (Constable, 2006) for nominally dry melt and olivine, respectively. Summary of results are listed in Table 3.1. Simulation results are reported as a function of the measured melt fraction (Fig. 3.4). A linear fit to the simulations results on a log-log scale, assuming that our data lie at the midpoint between our error bars (York et al., 2004), give Archie parameters $m=1.3\pm0.3$ and $A=0.66\pm0.06$ (Fig. 3.4). The value for our power-law exponent m agrees, within uncertainty, with data from Roberts and Tyburczy (1999) and ten Grotenhuis et al. (2005). Most of the differences in A between Roberts and Tyburczy (1999) and ten Grotenhuis et al. (2005) can be attributed to the different experimental condition.

3.3.2 Permeability

Laminar flow simulations were conducted on the same subvolumes as the direct current simulations. Permeabilities are plotted as a function of melt fraction and compared to bulk electrical conductivities (Fig. 3.4). A fit to the permeability data in log-log space gives power law exponent $n=2.7\pm0.7$, consistent with Miller et al. (2014). There is a clear difference in the power law curves between electrical conductivity and permeability. Fig. 3.5 shows that electricity flows more uniformly through the pore network and is less sensitive to pore diameters than fluid flow, which is consistent with the results of David (1993) and Martys and Garboczi (1992). Fluid flow, on the other hand, is dominated by a few major flow pathways, through which most of the mass is transported.

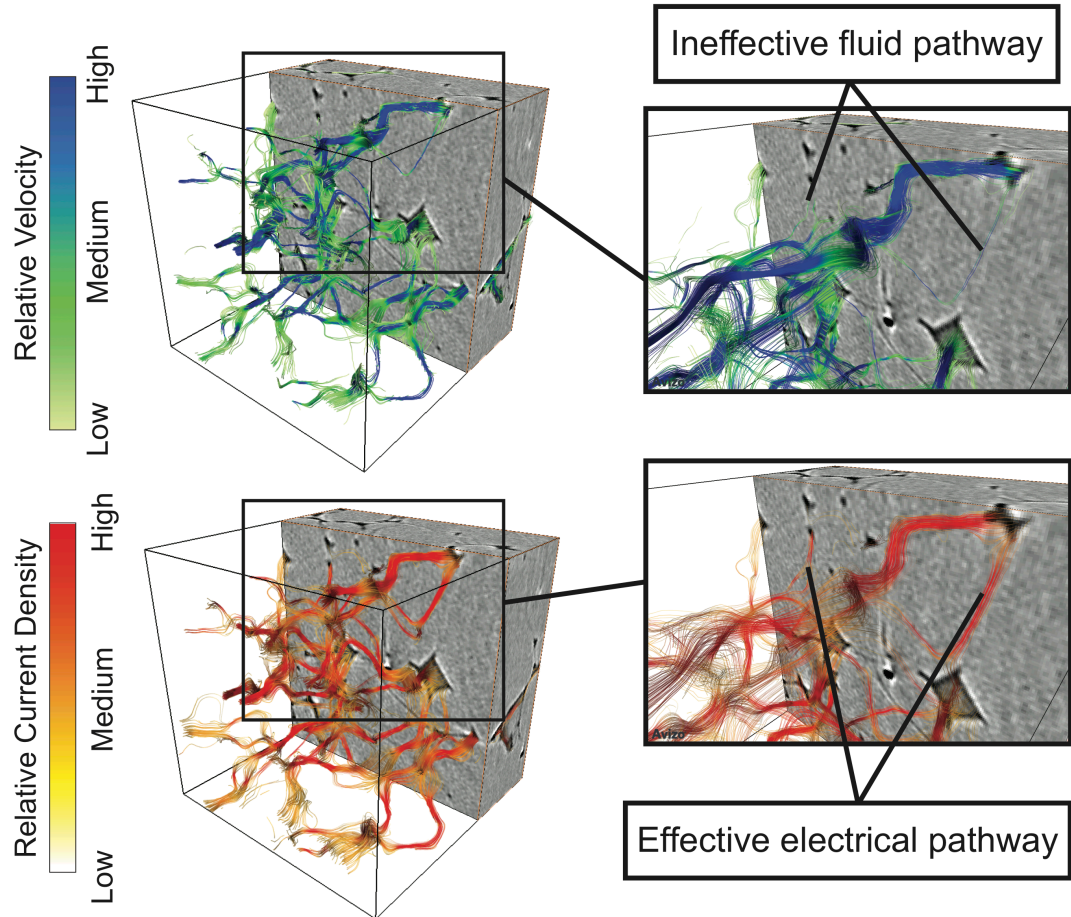


Figure 3.5: Visual comparison of fluid flow (top) and direct current (bottom) simulation results. Streamlines are computed for the velocity (fluid flow) and current density (direct current) fields from randomly placed in space occupied by melt. The locations of streamline seeds are determined randomly and weighted according to the magnitude of the field quantity to emphasize locations of high magnitude.

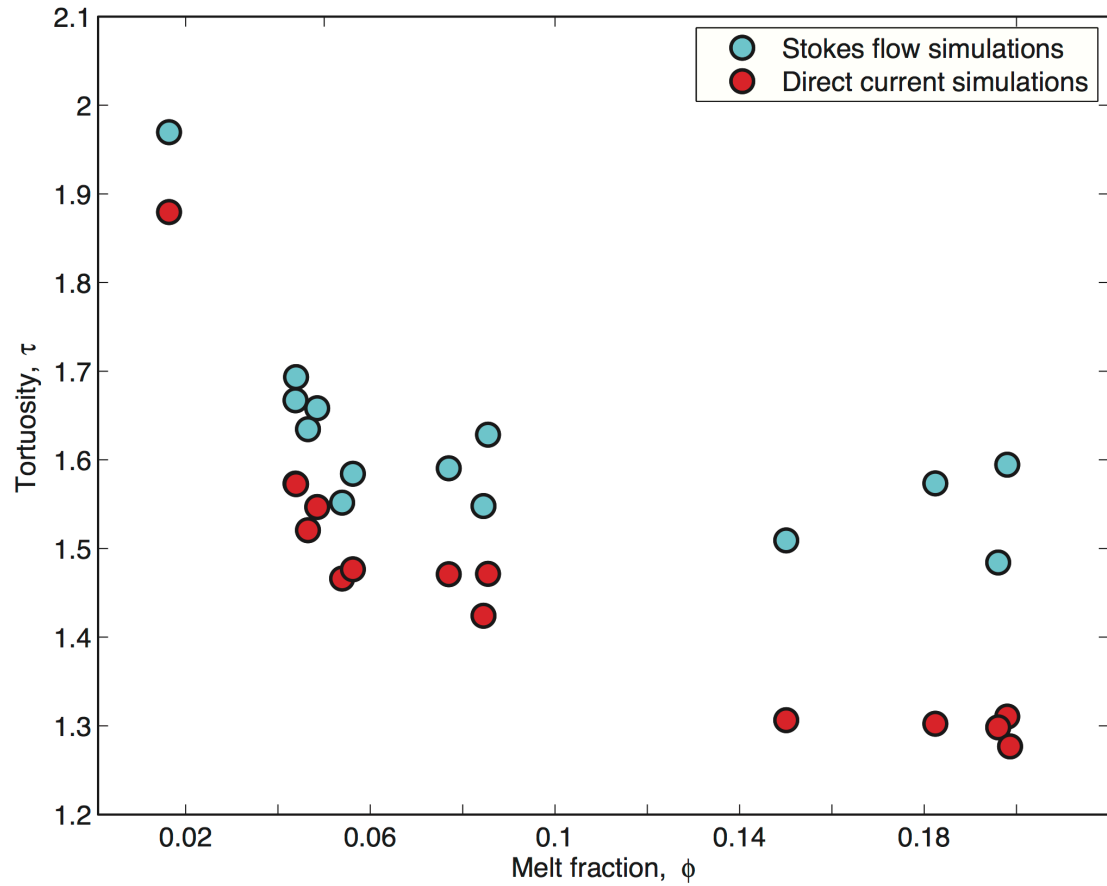


Figure 3.6: Comparison of tortuosity for fluid flow and direct current simulations. Olivine is assumed to be a perfect insulator by imposing a no-flux condition on the olivine-melt interface, so electricity is conducting through only melt.

3.3.3 Tortuosity

The tortuosity of our Stokes flow simulations, as computed by Eqn. (3.5) (Fig. 3.6), is consistently higher than direct current simulations, which provides quantitative evidence that electricity flows diffusively through the entire melt network, whereas fluid flow focuses along specific pathways. As fluid travels through distinctly different pathways through the melt network than does electricity, linking permeability to electrical conductivity is strictly empirical, with no microstructural justification. It should be noted that the high tortuosity of direct current and fluid flow simulations conducted at $\phi=0.02$ relative to those pertaining to higher melt fractions are likely due to low interconnectivity of the digital geometries $\phi \leq 0.02$.

3.4 Discussion

3.4.1 Electrical conductivity and permeability comparison

Differences between the permeability and electrical conductivity of an aggregate can be attributed to the differences in the radius dependence between the fluid and electric fluxes. Consider a simple network of interconnected tubes of various widths. For each tube, there is an analytical expression for the fluid and electric fluxes. The fluid flux (Q) is given by

$$Q = \frac{\pi b^4}{8\mu} \frac{\Delta P}{L} \quad (3.8)$$

where b is the radius of the tube, μ is the viscosity of the fluid, ΔP is the pressure difference from one end of the tube to the other, and L is the tube length. The electric flux (Φ) is given by

$$\Phi = \pi b^2 \frac{\Delta\Psi}{L} \quad (3.9)$$

where $\Delta\Psi$ is the difference in electric potential from one end of the tube to the other. The strong dependence of the fluid flux on the radius of the tube causes fluid flow to be far more sensitive to the pathways available to flow. Since mass and current are both conserved quantities, the strong radius dependence of fluid flux results in the formation of a so-called “critical pathway” (David, 1993) through which most of the material is transported. The fact that tortuosity for laminar flow simulations is consistently higher than direct current simulations is evidence of these critical pathways.

3.4.2 Comparison with experimental data

Our simulations results on electrical conductivity are compared to mixing models (Fig. 3.7A) that assume idealized melt geometries and electrical conductivities for each material. Five different idealized melt geometries are considered: parallel and series bounds, upper and lower Hashin-Shtrikman bounds (Hashin and Shtrikman, 1963), and a geometric average model. The parallel composite model, which assumes melt is organized into a series of pipes that extend, with zero tortuosity, through an insulating secondary phase, constitutes an absolute upper bound for the bulk electrical conductivity. The series model is the parallel model rotated 90°, so that current must pass through both olivine and melt. It is the absolute lower bound for the electrical conductivity of a composite material. Hashin-Shtrikman bounds are conceptually similar to the parallel and series bounds, except they assume a system composed of loosely packed spheres surrounded by a uniform

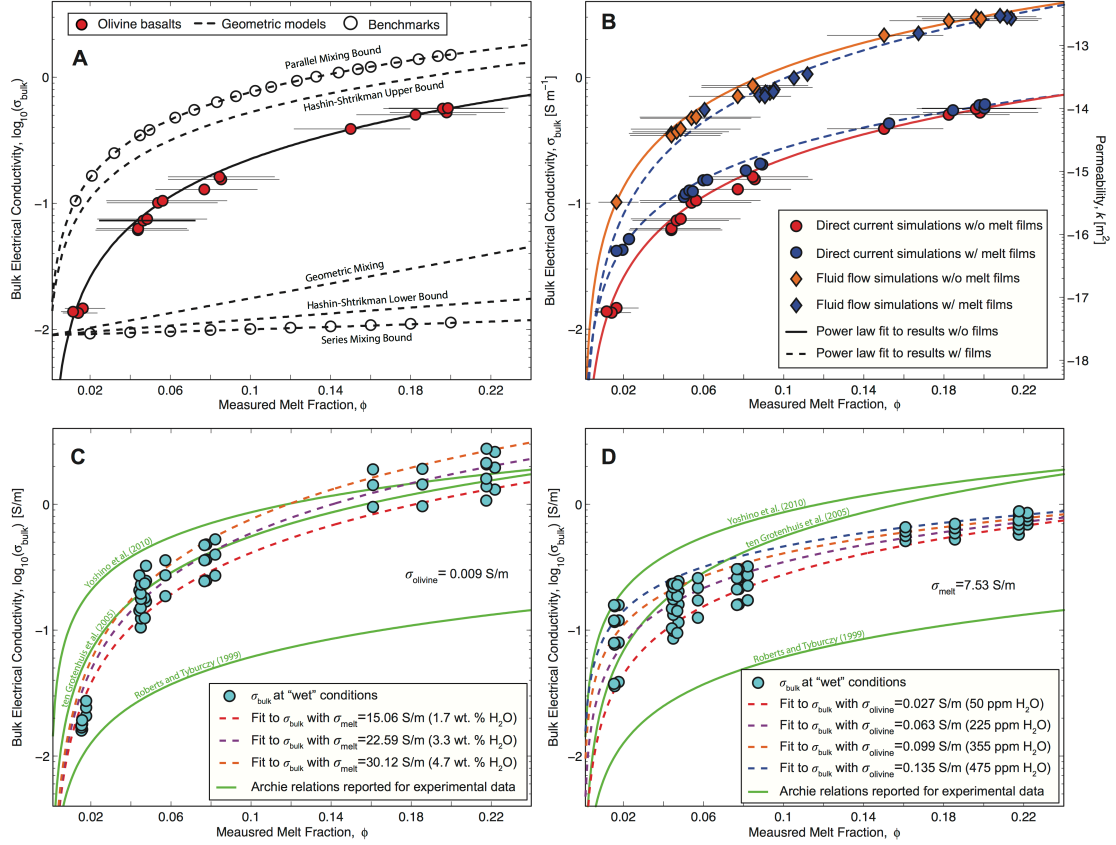


Figure 3.7: A) Comparison of bulk electrical conductivity computed from partially molten geometries obtain from μ -CT and mixing models for idealized melt geometries. Direct current simulations are conducted on synthetic datasets for straight tubes and serial layers for parallel and series bounds, respectively, to demonstrate the accuracy of the numerical model. B) Influence of melt films on bulk electrical conductivity and permeability computed using the three-phase conductivity model. C) Influence of H₂O in melt on the bulk electrical conductivity and comparison with experimental data. D) Influence of H₂O in olivine on the bulk electrical conductivity and comparison with experimental data.

layer of melt. The geometric mean model (Madden, 1976) considers a larger composite that consists of series and parallel sub-composites.

Fit to our simulation results yields power law parameters $m=1.3\pm0.3$ and $C=0.66\pm0.06$, which is between the Hashin-Shtrikman bounds, assuming $\sigma_{\text{olivine}}=0.009 \text{ S m}^{-1}$ and $\sigma_{\text{melt}}=7.53 \text{ S m}^{-1}$. This is consistent with a heterogeneous melt distribution like those observed in microscopy studies (e.g. Cmíral et al., 1998; Cooper and Kohlstedt, 1982; Faul and Fitz Gerald, 1999; Garapić et al., 2013; Miller et al., 2014).

Our results are systematically offset from the Hashin-Shtrikman upper boundary by a factor of 2 to 3. However, experimentally measured partially molten olivine-basalt electrical conductivities are often shown to overlap with the upper Hashin-Shtrikman bound (ten Grotenhuis et al, 2005; Yoshino et al, 2010). Although we do not have access to the actual samples from these studies, the chemistry, mineralogy, and preparation procedures are nominally the same as our own, suggesting that there is an additional contribution to the bulk conductivity that cannot be accounted for by separately considering the electrical conductivity of olivine and melt.

Recently, Zhang et al. (2014) measured the electrical conductivity of partially molten peridotite as a function of strain in simple shear. They separately measured the electrical conductivity of their melt and partially molten aggregates before and after deformation and found the electrical conductivity of their undeformed aggregate to be between the Hashin-Shtrikman lower and upper bounds, consistent with our study. Interestingly, the electrical conductivity of the deformed sample, measured in the

shear direction, overlapped the upper Hashin-Shtrikman bound. The change in conductivity before and after deformation may result from either a change in melt distribution or a reaction taking place during the experiment, which produces high conductivity phases. Similar effects may be present in other experimental studies in which measured values of electrical conductivity of partially molten samples are in agreement with the Hashin-Shtrikman upper bound.

Our study provides a rigorous link between melt distribution geometry and electrical conductivity. Direct current simulations on synthetic datasets of straight and parallel tubes are in good agreement with analytical solutions to the Laplace equation. Due to limitations in current imaging techniques, it is conceivable that some connections of the melt network are missing from the melt distribution obtained for samples with low melt fractions. However, the missing connections could not explain the discrepancy between our simulation results and the experimental data because simulations conducted on subvolumes containing nominal melt fractions 0.10 and 0.20, in which melt channels are completely interconnected, still yield electrical conductivity values less than experimental measurements. However, melt films, which are too thin to resolve with μ -CT, and the presence of H₂O in melt and olivine during electrical conductivity measurements, may play an important role. We describe below what the effect of these features would be.

3.4.3 Melt films

In addition to melt tubules and pools, a number of high-resolution studies (Cmíral et al., 1998; Faul, 2000; Faul et al., 1994; Garapić et al., 2013; Waff and

Faul, 1992; Wirth et al, 1996) document thin films of melt at some two-grain junctions. The thickness of melt films ranges between 3 nm and 100 nm. It has been suggested that thin films control melt transport at low melt fraction (Faul, 1997). We apply our direct current and Stokes flow models to quantifying the influence of melt films on bulk electrical conductivity and permeability using the approximations to melt films described in Section 3.2.2. By assuming all grain-grain boundaries are wetted by melt – i.e. the anisotropy of grain boundary surface energy is neglected—our approach gives an upper bound for the influence of melt films. Fluid flow simulations are conducted assuming that a full, 1-voxel fluid layer wets grain-grain boundaries.

Results are plotted in Fig. 3.7B. Artificially imposing melt films in our olivine-basalt geometry increases sample conductivity and has a similar effect on the bulk conductivity as increasing olivine electrical conductivity. This is because the voxels at grain boundaries are now considered an average of olivine and melt electrical conductivities, whereas these voxels were considered olivine only in the previous series of simulations. Bulk electrical conductivity increases substantially at low melt fraction and less so for higher melt fraction. The large error bounds on our melt fraction suggest that this change for $\phi > 0.02$ is within uncertainty. Nevertheless, the inclusion of melt films alone cannot account for the high electrical conductivities observed in experiments, even though their effect is grossly exaggerated in our simulations.

Including melt films substantially increases the permeability of our partial melt geometries. At the same time, the presence of melt films also substantially

increase melt fraction. The resulting porosity-permeability relationship does not differ significantly from that of Miller et al. (2014) without melt films. Actually, the permeability of a subvolume that includes melt films is systematically lower than permeability of a subvolume of similar porosity that does not include melt films. Although the difference is minimal and likely insignificant, melt films reduce permeability for a given porosity.

The larger effect of the melt films on bulk electrical conductivity relative to permeability is consistent with the concept of a critical pathway. In permeability computations, melt films contribute little to the critical pathways because fluid flux's strong dependence on hydraulic radius. In contrast, electrical conductivity flows more diffusively and uses melt films as viable pathways for electric transport. Thus melt films may be important contributors to the electrical properties of partially molten rocks, especially if their chemistry is distinct from the chemistry of the melt (Wirth, 1996). However, their contribution to the bulk electrical conductivity is not sufficient to account for the apparent discrepancy between the simulated and measured bulk electrical conductivities.

3.4.4 H₂O in melt

The presence of volatiles, specifically H₂O and CO₂, in melt is an excellent candidate for enhancing bulk electrical conductivity at high melt fraction. An addition of ~1 wt. % H₂O to an otherwise dry basaltic melt can increase the electrical conductivity by a factor of 3 (Ni et al., 2011). CO₂ has an even stronger effect on the melt electrical conductivity (Sifré et al., 2014; Yoshino et al., 2010) but is not

explicitly addressed here. To assess the effect of H₂O on the bulk electrical conductivity of partial molten rocks, we run direct current simulations for various melt conductivities and convert to H₂O concentration for the melt using

$$\log \sigma_{\text{melt}} = 2.172 - \frac{860.82 - 204.46\sqrt{C_{\text{H}_2\text{O}}}}{T - 1146.8} \quad (3.10)$$

where $C_{\text{H}_2\text{O}}$ is the concentration of H₂O in the melt and T is temperature. Starting values for melt electrical conductivity were adopted from measured values (ten Grotenhuis et al., 2005). Though different degrees of melting will likely produce subsequently different H₂O concentrations – since H₂O will partition from olivine to the melt – we assume a uniform increase in the melt conductivity.

Our results are presented in Fig. 3.7C. A H₂O concentration of 1.7 wt. % is sufficient to explain the high conductivities for high melt fraction in ten Grotenhuis et al. (2005) but underestimates the conductivity at lower melt fraction. Therefore, the electrical conductivity-melt fraction power-law does not match their experimental results at lower melt fraction. As melt fraction decreases, the electrical conductivity of olivine will have a stronger influence on the bulk electrical conductivity.

3.4.5 H₂O in olivine

Under hot, “dry” conditions, the electrical conductivity of olivine, which is controlled by polaron electron hopping (Constable, 2006; Dai et al., 2010; Schock et al., 1989; Wanamaker and Duba, 1993; Xu et al., 2000; Yoshino et al., 2009a), is three to four orders of magnitude less than that of basaltic melt and should contribute insignificantly to the bulk electrical conductivity. Under “wet” conditions, however, olivine electrical conductivity can increase significantly (e.g. Wang et al., 2006;

Yoshino et al., 2006, 2009; Poe et al., 2010; Jones et al., 2012; Dai and Karato, 2014a, 2014b; Gardés et al., 2014) though the magnitude of its influence on the bulk electrical conductivity of olivine is debated (Gardés et al., 2014). To explore the effect of an increased olivine electrical conductivity, we run direct current simulations using a range of higher olivine conductivities.

The conductivity of olivine with some fraction of water is estimated according to the model of Gardés et al. (2014). They consider three superposed conduction mechanisms. The first two, diffusion of cation vacancies and polaron hopping, operate under anhydrous conditions at high and low temperatures, respectively (Constable, 2006; Dai et al., 2010; Schock et al., 1989; Wanamaker and Duba, 1993; Xu et al., 2000; Yoshino et al., 2009a), while the third mechanism is related to the presence of hydrogen in olivine.

$$\sigma = \sigma_0^{\text{vacancy}} e^{-\frac{\Delta H^{\text{vacancy}}}{RT}} + \sigma_0^{\text{polaron}} e^{-\frac{\Delta H^{\text{polaron}}}{RT}} + \sigma_0^{\text{hydrous}} C_{\text{H}_2\text{O}} e^{-\frac{\Delta H^{\text{hydrous}} - \alpha C_{\text{H}_2\text{O}}^{1/3}}{RT}} \quad (3.11)$$

where ΔH are the activation enthalpies for the mechanisms, $C_{\text{H}_2\text{O}}$ is the weight concentration of H_2O in the olivine, α corrects for the decrease in the activation enthalpy for increasing H_2O concentration, R is the ideal gas constant, and T is temperature.

Results are plotted in Fig. 3.7D. We find that increasing olivine conductivity noticeably enhances the bulk conductivity at low melt fraction and changes the shape of the bulk electrical conductivity-melt fraction power-law. If we assume wet conditions for both olivine and melt, we find that $\sigma_{\text{melt}}=15.60$ S/m and $\sigma_{\text{olivine}}=0.045$ S/m explains experimental data by ten Grotenhuis (2005). An olivine electrical conductivity of 0.045 S/m at 1475 °C translates to ~145 ppm. Given, the solubility of

H₂O in olivine ~90 ppm (Gaetani et al., 2014), measured at 1200 °C, it is difficult to justify 145 ppm H₂O concentration in olivine. However, without solubility data measured at higher melt temperature, it is unclear whether the solubility of H₂O in olivine at 1200 °C can be extrapolated to 1475 °C.

The trend of the data from laboratory measurements (e.g., ten Grotenhuis et al., 2005) may reflect water in the aggregates, with the combined effect of water in olivine and melt films dominant at low melt fraction and water in the melt dominant at high melt fraction. Neither effect is expected to significantly affect the relation between permeability and melt fraction.

3.4.6 Chemical heterogeneity

We speculate the existence of a thin, electrochemically distinct layer at the olivine-melt interface that might account for the apparent discrepancy between the bulk electrical conductivities measured and those we computed using real partial melt geometries. Electrolytic conduction by Na⁺ ions dissolved in the fluid is the primary mode of electrical transport in porous sedimentary rocks (Nover, 2005). High concentration of Na⁺ ions at the mineral-fluid interface would provide an additional pathway for electrical conduction. In crustal rocks, lattice deficiencies at the surface of clay minerals result in a locally negative charge that attracts Na⁺ (Nover, 2005), coating the mineral-fluid interface with a thin, highly conductive layer, often called the electric double layer (EDL). The thickness of the EDL is roughly the Debye length (Debye and Hückel, 1923; Morgan et al., 1989), which depends on physical parameters of the fluid phase, such as the molarity and permittivity of solution. For

reference, the Debye length of the clay-water interface is on the order of a few to tens of nanometers (Tombácz and Szekeres, 2006; Wan and Tokunaga, 2002). Though the thickness of the EDL is quite small compared to the diameter of the melt conduits, the local electrical conductivity of the EDL would be greater than that of the fluid, and since it forms an interconnected pathway, will conduct in parallel with the fluid. Therefore, electrical conduction near the mineral-fluid interface may be a separate and important conduction mechanism to consider, especially at low fluid fraction.

The existence of EDLs in partially molten olivine-basalts is currently not considered, since the chemistry of olivine-melt interface is intrinsically different from the clay-water interface. The formation of an EDL on an olivine-melt interface would require a local charge imbalance, possibly due to concentration of impurities. Gurmani et al. (2011) and Wirth (1996) have proposed chemical variations in the presence of melt films but not for every olivine-melt interface. Nevertheless, the presence of EDLs – or more generally a spatial heterogeneity of the primary charge carriers – is a convenient mechanism to reconcile our model results and laboratory measurements. Furthermore, the influence of EDL on the bulk conductivity may be invisible to impedance spectroscopy if electrical conduction through EDLs operates in the same frequency spectrum as electrolytic conduction. Unfortunately, modeling the influence of the EDL on bulk conductivity requires a priori knowledge of the Debye length and EDL electrical conductivity. These variables, to our knowledge, have not been constrained for the partially molten olivine-basalt system.

3.5 Conclusion

We modeled direct current on experimentally obtained olivine-basalt partial melt geometries in order to link microstructural properties to bulk electrical conductivity and deconvolute the role of melt geometry from other processes, e.g. volatile content, that may affect electrical properties. Our digital rock physics approach for determining the bulk electrical conductivity of partially molten rocks has the benefit of having fine control on the physics and material properties of the system, while still adhering to a real melt geometry. Rather than having to rely on an idealized melt geometry from measured electrical properties of the system, we are able to compute electrical properties directly from the melt microstructure.

We found that the high bulk electrical conductivities observed in experiments cannot be accounted for by considering only a two-phase olivine-melt model unless there is significant enhancement of the melt electrical conductivity by volatiles. The trends observed in laboratory measurements may reflect water in the aggregates, with the combined effect of water in olivine and melt films dominant at low melt fraction, and water in the melt dominant at high melt fraction. Neither effect is expected to significantly affect the relation between permeability and melt fraction. We speculate that a high electrical conductivity, chemically distinct electrochemical layer on the grain-melt interface may also affect laboratory measurements. Such layers have been well characterized in rocks that contain clay minerals but have not been discussed in the context of partially molten mantle rocks.

Chapter 4: Experimental evidence for lithologic melt partitioning between olivine and orthopyroxene in partially molten harzburgite

Abstract

The grain-scale distribution of melt in partially molten aggregates under isostatic stress is controlled by gradients in surface energy associated with the grain-grain and grain-melt boundaries. For a monomineralic aggregate, e.g. olivine-basaltic melt composed of idealized isotropic grains, melt is more or less equally distributed among grains. However, in a polymineralic aggregate, e.g. olivine-orthopyroxene (opx)-basaltic melt, spatial variations in surface energy cause melt to partition unevenly among the mineral components in favor of a lower energy configuration. In an aggregate that has substantial mineralogical variability, this phenomenon, known as lithologic fluid partitioning, can act as a mechanism for concentrating melt and possibly modify permeability and electrical conductivity of the rock.

Experimental studies that examine analogue systems, e.g. calcite-fluorite- H_2O , observe strong fluid partitioning among the constituent minerals. However, experimental evidence for melt partitioning between olivine and opx, the two most relevant minerals to the upper mantle, is lacking. We present experimental results that elucidate the degree of melt partitioning between olivine and opx in partially molten harzburgites.

Samples were prepared by mixing powdered oxides and natural, high-alumina basalt in various proportions to test for lithologic melt partitioning across a range of melt fractions. Bulk composition was such that a 3 to 2 olivine to opx ratio was

maintained over all samples; though the measured olivine to opx ratio for subvolumes varies widely 1.2 and 4.3 between subvolumes. Samples were cored and imaged using synchrotron-based X-ray micro-computed tomography, producing a high-quality three-dimensional digital sample. Representative subvolumes were cropped from the digital samples, avoiding decompression fractures where possible. Grayscale subvolumes were transformed into label images whereby each voxel is assigned a phase identification number, e.g. 1 for melt, 2 for olivine, and 3 for opx. Local melt fraction distributions for olivine and opx were automatically characterized for each subvolume by counting voxels inside ellipsoidal envelopes that were fitted to each olivine and opx grain, respectively.

We find that melt partitions in a 1.1 to 1.5 ratio between olivine and opx, respectively, across all subvolumes. We present lithologic melt partitioning as a mechanism for focusing melt in the mantle that could potentially enhance average melt ascent velocities.

4.1 Introduction

Melt transport at mid-ocean ridges is thought to operate via porous flow along an interconnected, intergranular network (Turcotte and Schubert, 2014). Geochemical data collected from mid-ocean ridge basalt suggest melt flux is likely dominated by melt fraction heterogeneities that are larger than the grain size. For example, secular disequilibrium of uranium-series nuclides (Condomines et al., 1981; Iwamori, 1994; Newman et al., 1983; Volpe and Hammond, 1991) and the undersaturation of opx with respect to olivine (Kelemen et al., 1997) are indirect evidence of high-melt

fraction, high-permeability conduits. Two mechanisms of interest have been proposed to organize melt on length scales comparable to the compaction length – a natural length scale that depends only on the material properties of the partially molten mantle rock: the reaction infiltration instability (RII) (e.g. Aharonov et al., 1995; Daines and Kohlstedt, 1994; Kelemen et al., 1995a; Spiegelman et al., 2001) and deformation-induced melt segregation (e.g. Holtzman and Kohlstedt, 2007). The former is a consequence of the positive feedback between melt flux and opx dissolution, and the latter results from the anisotropic viscosity of partially molten rock (Qi et al., 2014; Allwright and Katz, 2014). We propose an additional mechanism that can concentrate and organize melt: lithologic melt partitioning, which is a consequence of the thermodynamic gradient caused by spatial variations in mineralogy.

At equilibrium, melt distributes into a configuration that minimizes the total surface energy of the system. An idealized system composed of uniform, isotropic olivine grains, the minimum-energy configuration is one in which the melt fraction is the same around every grain (Fig. 4.1A). However, the presence of secondary mineral, such as orthopyroxene (opx), which has a higher solid-melt surface energy density than olivine, will perturb the uniform surface energy distribution, causing melt to concentrate in olivine-rich regions. This phenomenon, known as lithologic melt partitioning, where melt partitions unevenly between olivine and opx, results in a locally high melt fraction in olivine-rich regions and a locally low melt fraction opx-rich regions.

An alternative – but equivalent – pedagogical model for understanding

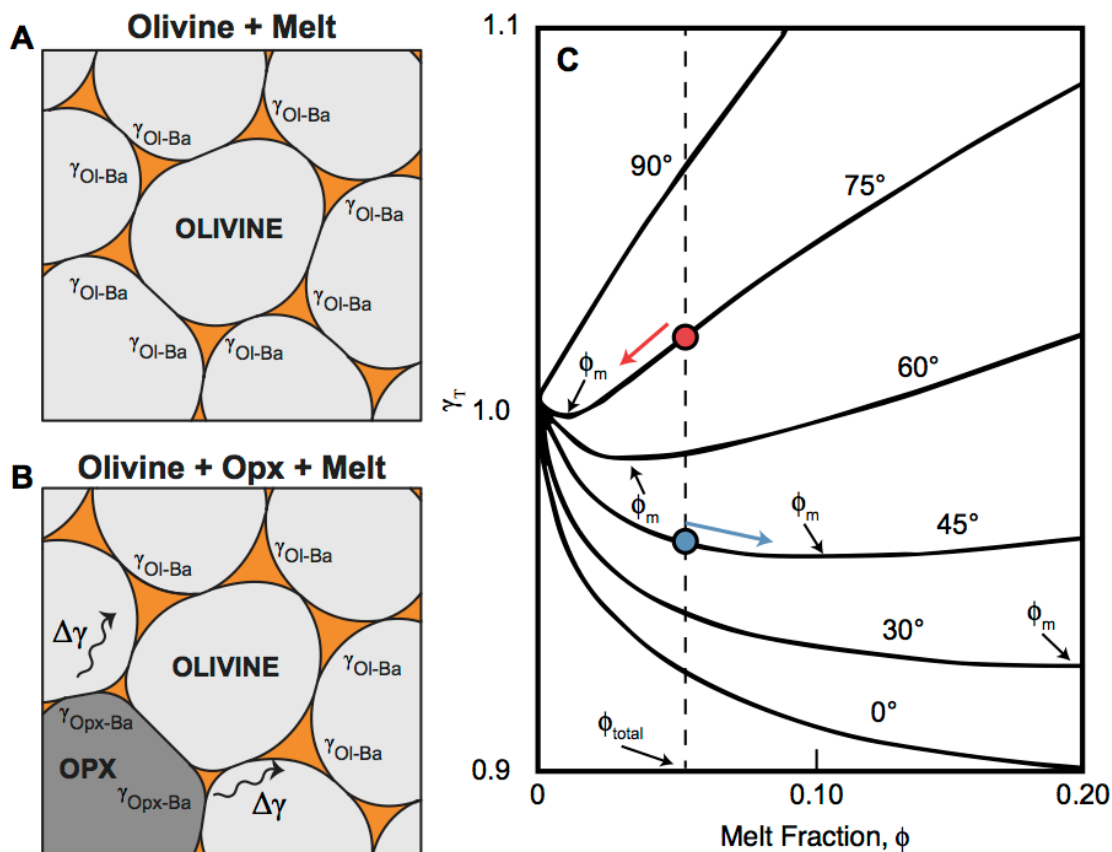


Figure 4.1: Schematic diagram illustrating the minimum energy melt fractions for olivine and opx in a close system containing a finite amount of melt. Modified from (Park and Yoon, 1985; Watson, 1999). (A) Schematic diagram of an aggregate containing only olivine and basaltic melt. Local melt fraction is the same for every grain. (B) Schematic diagram of an aggregate containing olivine, opx, and basaltic melt. Spatial variations in the surface energy distribution, resulting from different mineral component, cause the basaltic melt to partition unevenly between olivine (higher melt fraction) and opx (lower melt fraction). (C) Total surface energy contained in a melt-bearing rock system normalized by the total surface energy contained in a melt-free system plotted as a function of melt fraction for various dihedral angles. Dotted line represents the total melt fraction contained in an example system. In a homogeneously mixed olivine-opx-basaltic melt aggregate, local melt fractions associated with olivine (blue circle) and opx (red circle) will adjust to minimize the energy of the system. Arrows with ϕ_m indicate the minimum-energy melt fraction for an open system that is exposed to an infinite melt reservoir.

lithologic melt partitioning uses the concept of the minimum-energy melt fraction. For a given dihedral angle (Eqn. 1.1), there is a melt fraction, called the minimum-energy melt fraction (Fig. 4.1C), that minimizes the total interfacial energy of the system. Consider a simple system consisting of a monomineralic aggregate that is open to a melt reservoir. The aggregate will draw melt from the reservoir via capillary action until the minimum energy melt fraction is attained. However, in the upper mantle, olivine and opx grains coexist. In this more realistic scenario, melt will partition unevenly between olivine and opx but will not attain their nominal minimum-energy melt fractions for the given dihedral angle.

Lithologic melt partitioning was observed in analogue systems that consisted of two juxtaposed mineral aggregates and interstitial H₂O. For example, piston-cylinder experiments (Watson, 1999) showed that H₂O partitions in a 5 to 2 ratio between fluorite and quartz, respectively, and in a 3 to 1 ratio between clinopyroxene and quartz (Fig. 4.2), respectively. In the same study, lithologic melt partitioning was examined using juxtaposed olivine and opx aggregates containing the same initial proportions of basaltic melt. Since the surface energy density of the olivine-basaltic melt interface is markedly lower than that of the opx-basaltic melt interface, it is surprising that the sample exhibited no measurable melt partitioning. Watson concluded that the distance separating the olivine and opx-rich regions was too large (a few millimeters) and viscosity of basaltic melt was too high for lithologic melt partitioning to occur in the timeframe of the experiment (~6 days).

We approach measuring lithologic melt partitioning in partially molten harzburgitic rocks using a novel approach. Experimental charges, composed of

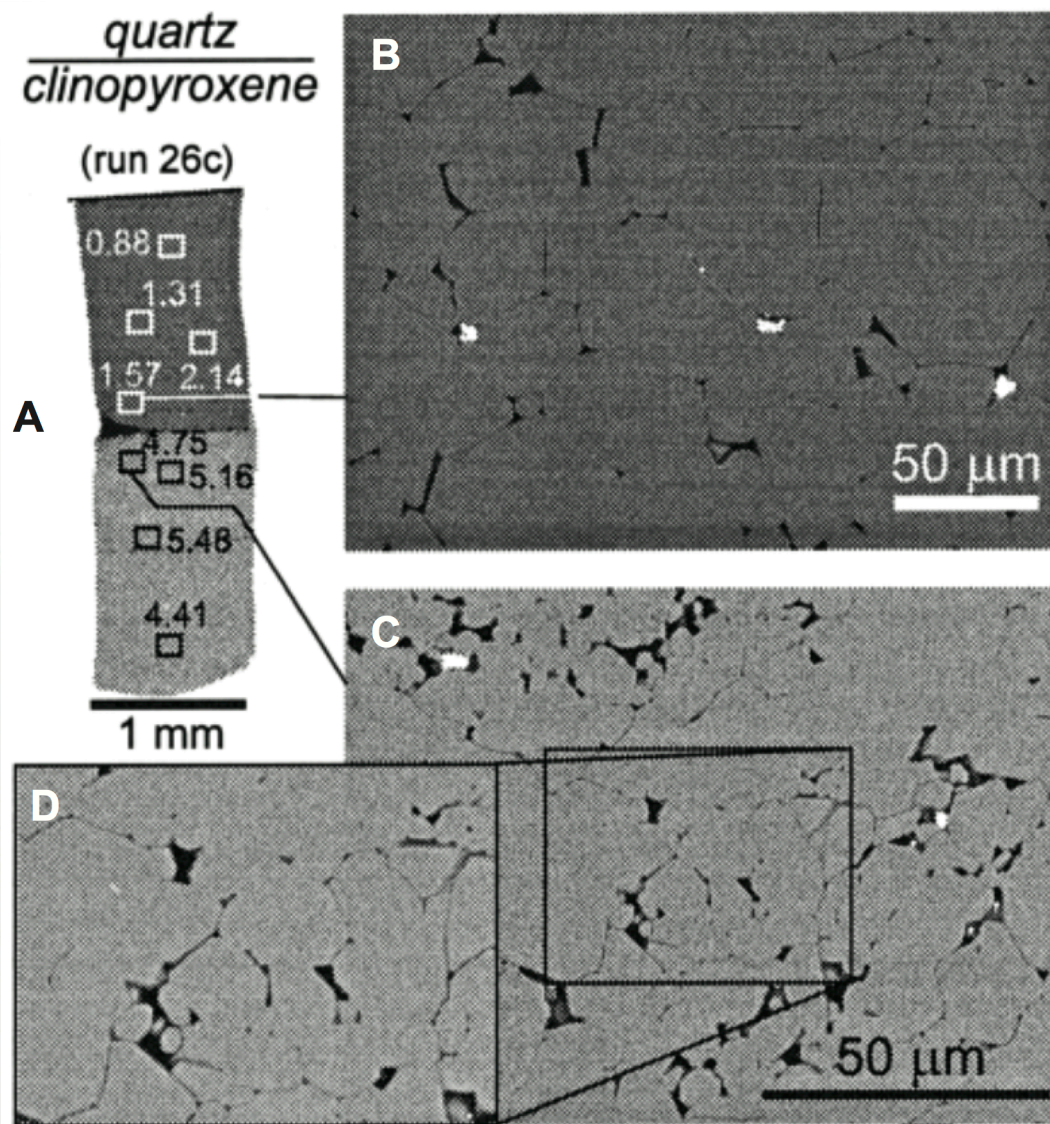


Figure 4.2: Fluid distribution in cylindrical sample consisting of juxtaposed quartz and clinopyroxene (Watson, 1999). (A) Back-scattered electron (BSE) image of a polished cross-section. White and black boxes denote areas where melt fractions were measured. (B) BSE image of one of the quartz analysis locations. (C) BSE image of one of the clinopyroxene analysis locations. (D) Close-up of the clinopyroxene-H₂O microstructure.

various proportions of olivine, opx, and basaltic melt, were synthesized in solid-media piston-cylinders apparatuses. Olivine and opx grains were homogeneously mixed, which reduced the length-scale of partitioning three orders of magnitude. Cores were drilled from the samples and imaged in three-dimensions (3-D) using synchrotron-based X-ray micro-computed tomography (μ -CT). Statistically representative volumes were cropped from each sample and local melt distributions were obtained for olivine and opx by systematically measuring the proportion of melt in each olivine and opx-rich region.

4.2 Methods

4.2.1 Sample preparation of harzburgite samples

Harzburgite samples were prepared by hot, isostatic pressing of a mixture containing oxides and natural, high-alumina basalt. The oxide mixture was prepared by homogenizing oxides mixed in proportion such that olivine (forsterite) and opx (enstatite) crystals would have the same chemistry as those found in a natural harzburgite collected from the Southwest Indian Ridge (Dick, 1989). For each melt fraction, the oxide proportions were adjusted to maintain a nominal 3 to 2 (olivine to opx) ratio. The ingredients and chemical proportions used in making the oxide mixtures are reported in Table 4.1.

Not all of the elements could be added to the mix as oxides. Calcium, for example, was added in carbonate form (CaCO_3). The mix was homogenized for six one-hour cycles using an automatic agate mortar and pestle. Upon completion, we applied a decarbonation procedure to transform the carbonates to oxides. To

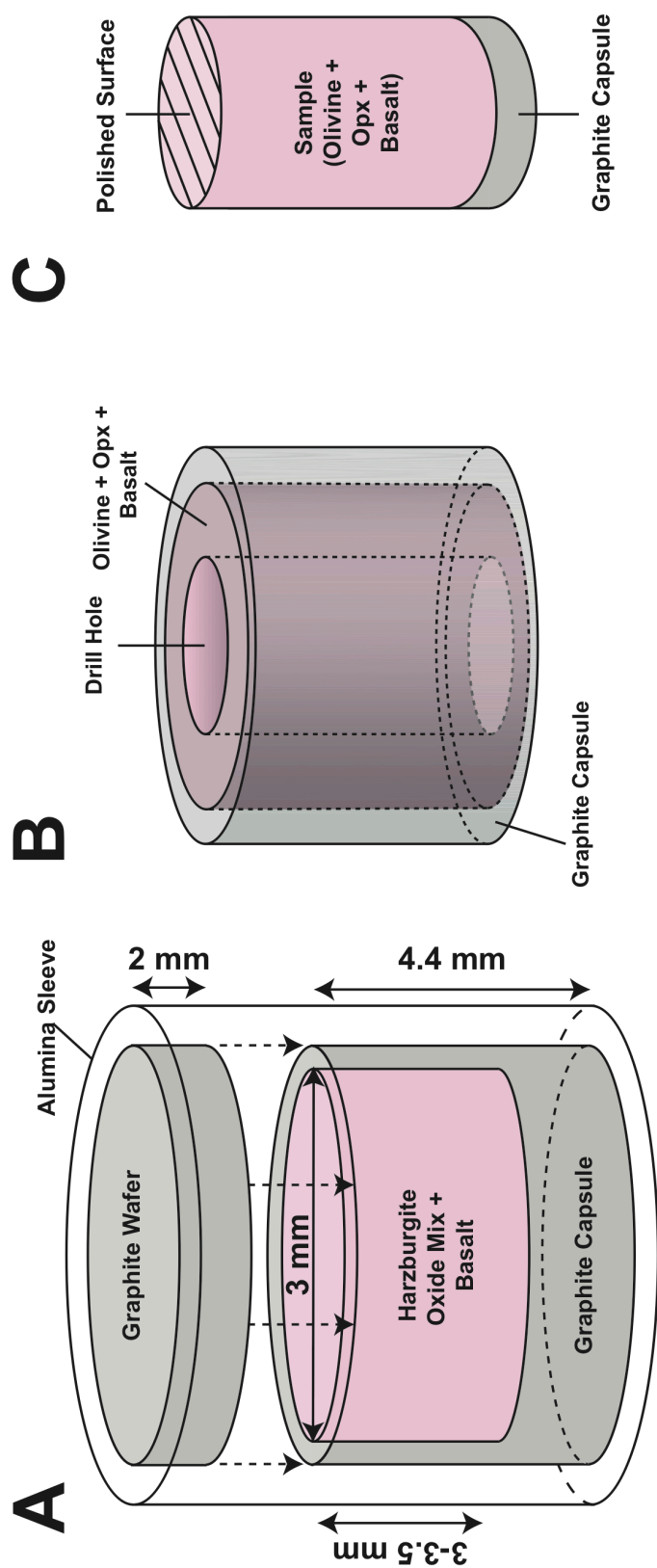


Figure 4.3: Schematic diagram of experimental setup before and after sintering. (A) Starting materials before sintering. The graphite capsule (gray), which is packed with the powdered olivine/basalt mixture (pink), is placed in an alumina sleeve with a graphite wafer as a lid. (B) Sample assembly after sintering. One side of the sample assembly is cut and the surface of the sintered aggregate is polished. The hole in the sample assembly illustrates where the sample (C) is taken for X-ray imaging. Hash marks represent the polished surface.

Component	Drying Conditions	Measured Weight (g)
SiO ₂	28 hours @ 1000 °C	5.03010
TiO ₂	N/A	0.00090
Al ₂ O ₃	28 hours @ 1000 °C	0.13627
Fe ₂ O ₃	1 hour @ 800 °C	0.91690
MnO ₂	2 hours @ 800 °C	0.01649
MgO	216 hours @ 1000 °C	4.65014
CaCO ₃	4 hours @ 400 °C	0.16765
Na ₂ CO ₃	N/A	0.00074
K ₂ CO ₃	N/A	0.00077
NiO	2 hours @ 800 °C	0.01961

Table 4.1: Starting composition and drying conditions of harzburgite oxide/carbonate mix. Mixed in proportion to attain a 2 to 3 olivine to opx harzburgite mix.

decarbonate the mixture, it was placed in a furnace at 300 °C and heated to 850 °C at 100 °C/hr. The mix was held at 850 °C for a minimum of 24 hours. Pulverized natural basalt was added in various proportions to the oxide mix to attain total melt fractions of 0.02, 0.05, 0.10, and 0.20 when melted. The same homogenization procedure was repeated for every oxide-basalt mixture.

For each melt fraction, ~36 mg of the oxide-basalt mixture was cold-pressed into a cylindrical pellet using a 1-ton press and placed into a graphite capsule (Fig. 4.3). Capsules were dried overnight at 400 °C to remove surface H₂O from the experimental charges. Charges were placed in solid-medium piston-cylinder apparatuses and brought up to 1.5 GPa and 1350 °C using the cold piston-in technique (Johannes et al., 1971). Details about the uncertainty in pressure and temperature can be found in Chapter 2.2 and Appendix A.

Upon completion of the piston-cylinder runs, experimental charges were quenched by turning off the heating source while maintaining a steady flow of cold water through the space surrounding the pressure vessel. Cylindrical 1-mm cores were drilled from each sample along the cylindrical sample axis (Fig. 4.3).

4.2.2 Imaging procedure

The image acquisition, pre-processing, and data reduction procedures are outlined in Fig. 4.4.

Cylindrical harzburgite samples were imaged using a synchrotron light source at beamline station 2BM-a of the Advanced Photon Source, Argonne National Laboratory. The very small density contrasts at olivine-opx and olivine-basalt

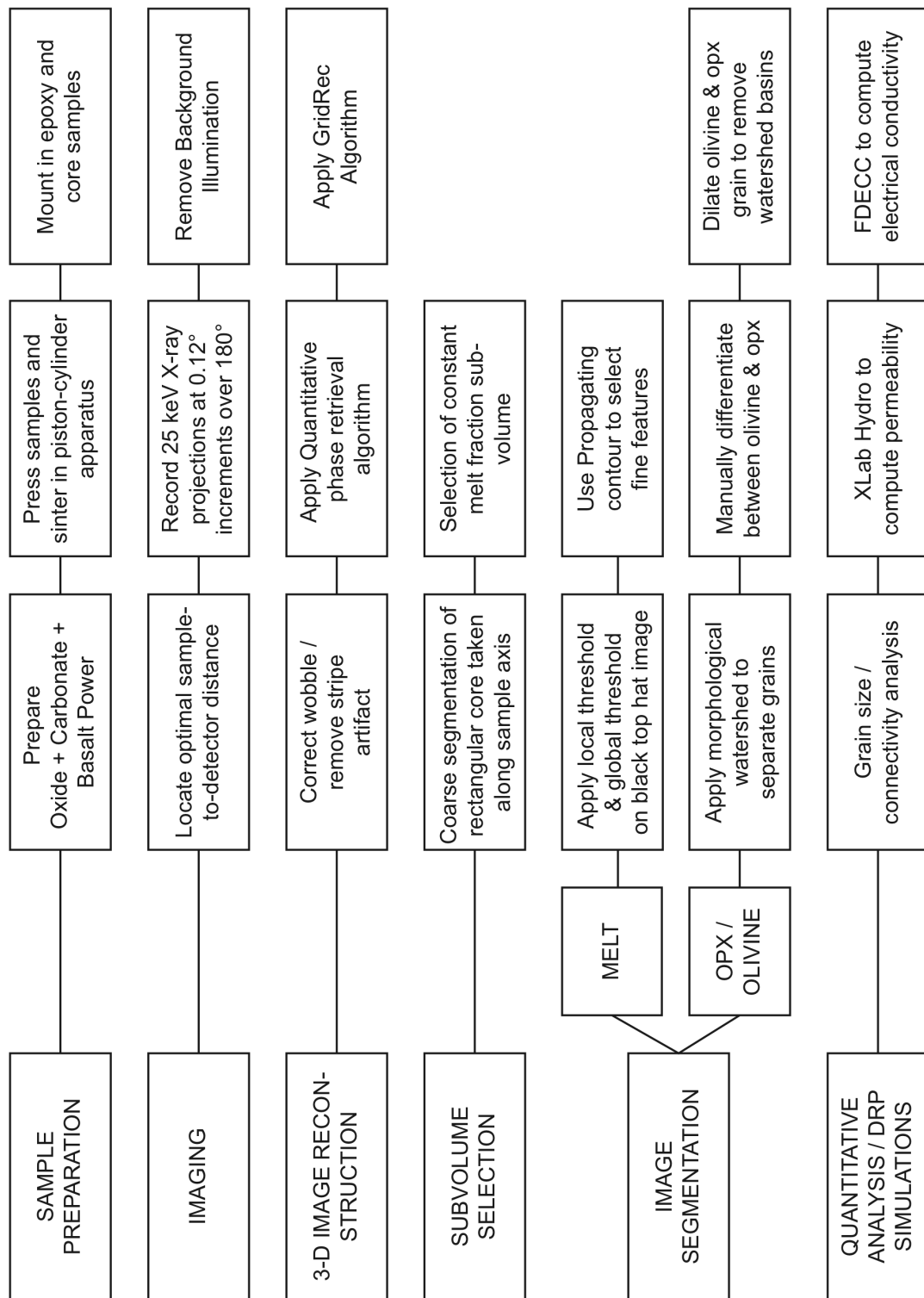


Figure 4.4: Workflow for digital rock physics technique, starting at sample preparation and ending with quantitative analysis and simulation.

boundaries warranted a novel imaging procedure, which involved a combination of absorption-contrast and phase-contrast imaging techniques. A monochromator was used to select a narrow energy spectrum around 24.4 keV. The sample was rotated 180° through the X-ray beam, and at every 0.12° increment, we recorded a snapshot of the X-ray projection using a CCD camera. Each projection contains information about the X-ray absorption and phase integrated along the trajectory of the X-ray. Prior to reconstruction, the background illumination was removed from each projection.

The open source, Python-based software Tomopy (Gürsoy et al., 2014), which was developed by the beamline scientists at Advanced Photon Source, was used to perform the image reconstruction. First, a stripe-removal algorithm based on Münch et al. (2009) was applied. A quantitative phase retrieval algorithm, which was based on Paganin et al. (2002) was used to simultaneously recover the X-ray absorption and diffraction signal. Finally, GridRec (Dowd et al., 1999) was used to perform the tomographic reconstruction. In the resulting grayscale image (Fig. 4.5, olivine (lightest granular phase), opx (darkest granular phase), and quenched basaltic melt (dark interstitial phase) are clearly distinguishable.

4.2.3 Subvolume Selection

A visual inspection of the whole sample reconstruction reveals strong melt fraction heterogeneity along the cylindrical axis of each sample (Fig. 4.6). The melt fraction is at a minimum at the bottom of sample and a maximum at the top of the sample. We interpret this long-wavelength heterogeneity to be caused by melt

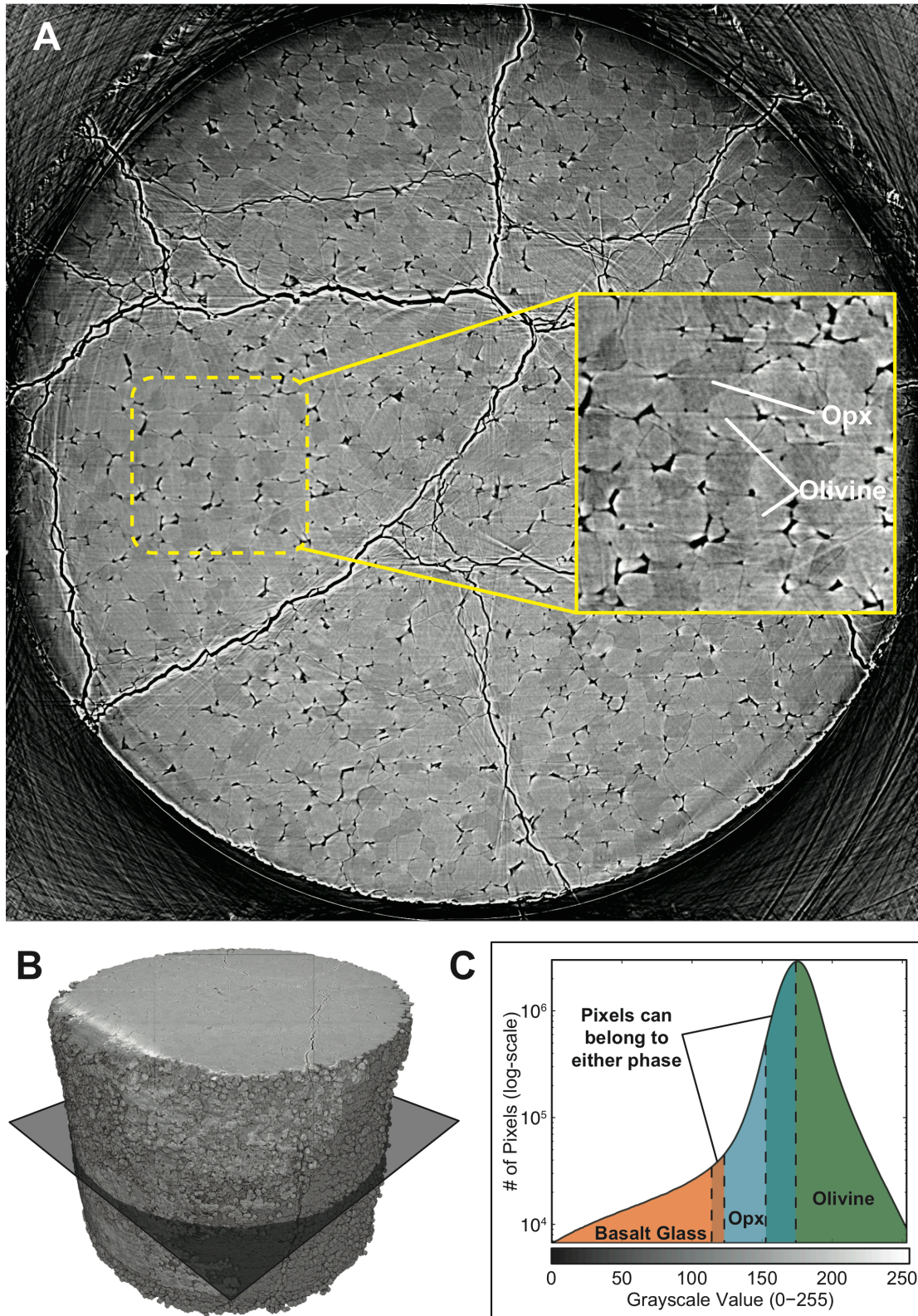


Figure 4.5: Visualizing the X-ray μ -CT data. (A) Tomography slice sampled at depth from harzburgite samples. Samples contains ~ 5 vol. % quenched basaltic melt (glass). On the right is a closeup of the tomography slice. Olivine (light granular phase), opx (dark granular phase), and basaltic glass (dark interstitial phase) are clearly visible. (B) View of whole sample. Dark plane represents the location of the displayed tomography slice. (C) Histogram of a subvolume cropped from the sample.

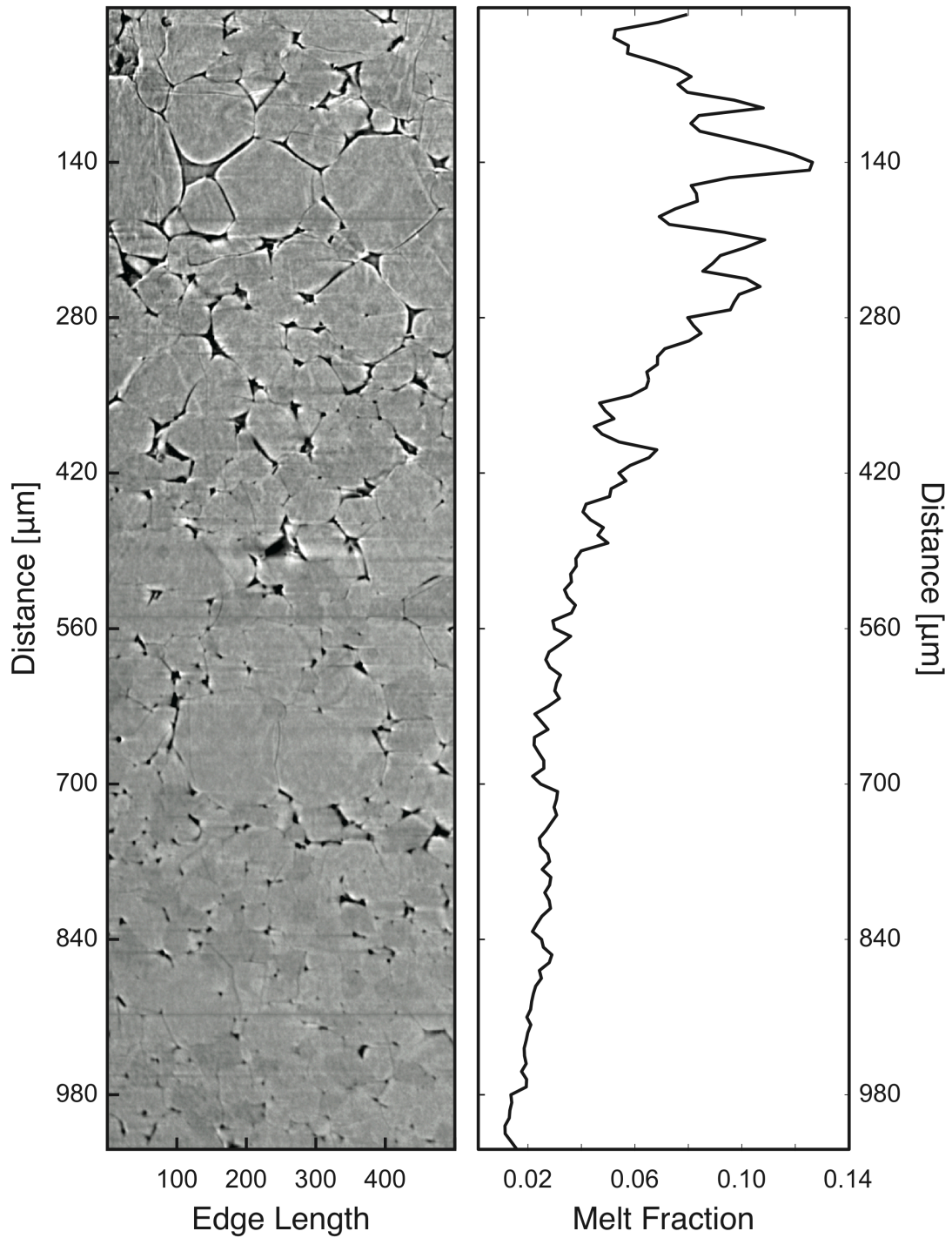


Figure 4.6: Vertical melt fraction heterogeneity. Orientation of the sample during sintering is the same as the orientation of the image. (Left) Grayscale image taken along the vertical axis of the sample. (Right) Melt fraction measured over a over 10-slice intervals. Melt fraction is highest at top of sample and lowest at the bottom of the sample.

buoyancy. Over the course an experiment, melt which is less dense than the surrounding olivine or opx, will rise to the top of the sample. As the melt rises, the loss of mass towards the bottom of the sample is compensated by compaction of the granular matrix. As melt fraction decreases at the bottom of the sample, surface tension and compaction forces reach mechanical equilibrium with buoyancy.

Following Miller et al. (2014) and Watson and Roberts (1999), smaller subsets of data, which we call subvolumes, were cropped from each reconstructed 3-D image at locations of relatively constant melt fraction. Decompression fractures (Fig. 4.5A) and long-wave-length melt fraction heterogeneity (e.g. Fig. 4.6B between 280 and 560 μm) were avoided.

4.2.4 Image segmentation

In order to characterize the melt distribution, each grayscale subvolume was converted to a label image: grayscale voxels were assigned values 1, 2, or 3 for basaltic glass, olivine, or opx, respectively. We developed a semi-automatic segmentation workflow. First, a trial segmentation of the melt is performed using a combination of Avizo's local thresholding module and tophat global threshold. Thin decompression fractures are manually removed from the image by overlapping the trial segmentation with a morphological erosion and dilation of the image using a $2 \times 2 \times 2$ voxel³ ball-shaped kernel. Avizo's morphological filter toolbox was used.

Subtle contrast at the olivine-opx interfaces and bright imaging artifacts at the grain edges prevented us from applying the same local threshold technique to differentiate the opx from the olivine. We used a morphological watershed

transformation to separate grains and then handpicked opx grains from the aggregate. Grains that were incorrectly separated were corrected using Avizo's propagating contour tool. Once all of the opx grains were differentiated from olivine, the watershed basins were removed by a simultaneous dilation of the olivine and opx images. To remove jagged edges, which are artifacts of the morphological watershed tool, the resulting image was smoothed using an $3 \times 3 \times 3$ voxel³ Gaussian kernel. The segmented melt image is superposed on the olivine-opx segmented image. The result is a very accurate segmentation of the melt and a slightly less accurate approximation of the olivine-opx grain boundaries. Accurately estimating the location of the olivine-melt and opx-melt interfaces is far more important than the olivine-opx interface since we are most interested in the local melt fraction around each grain. 3-D volume renderings of the label images are given in Fig. 4.7.

In some samples, a bright, dendritic phase, which we think is partially recrystallized melt, appears in the melt near the olivine-melt interface. Partially recrystallization of the melt is usually indicative of an imperfect quench. Since they are not present melt prior to quenching, voxels associated with these dendritic features are assigned to melt in the segmentation procedure. Refer to Appendix C, Fig. C.1) for an image of the bright dendritic phase.

4.2.5 Quantification of local melt fraction distribution

Though a homogeneous mixture of olivine and opx reduces the amount of time required to reach a steady state microstructure, it complicates evaluation of the characteristic melt fraction associated with each mineral phase. Therefore, we

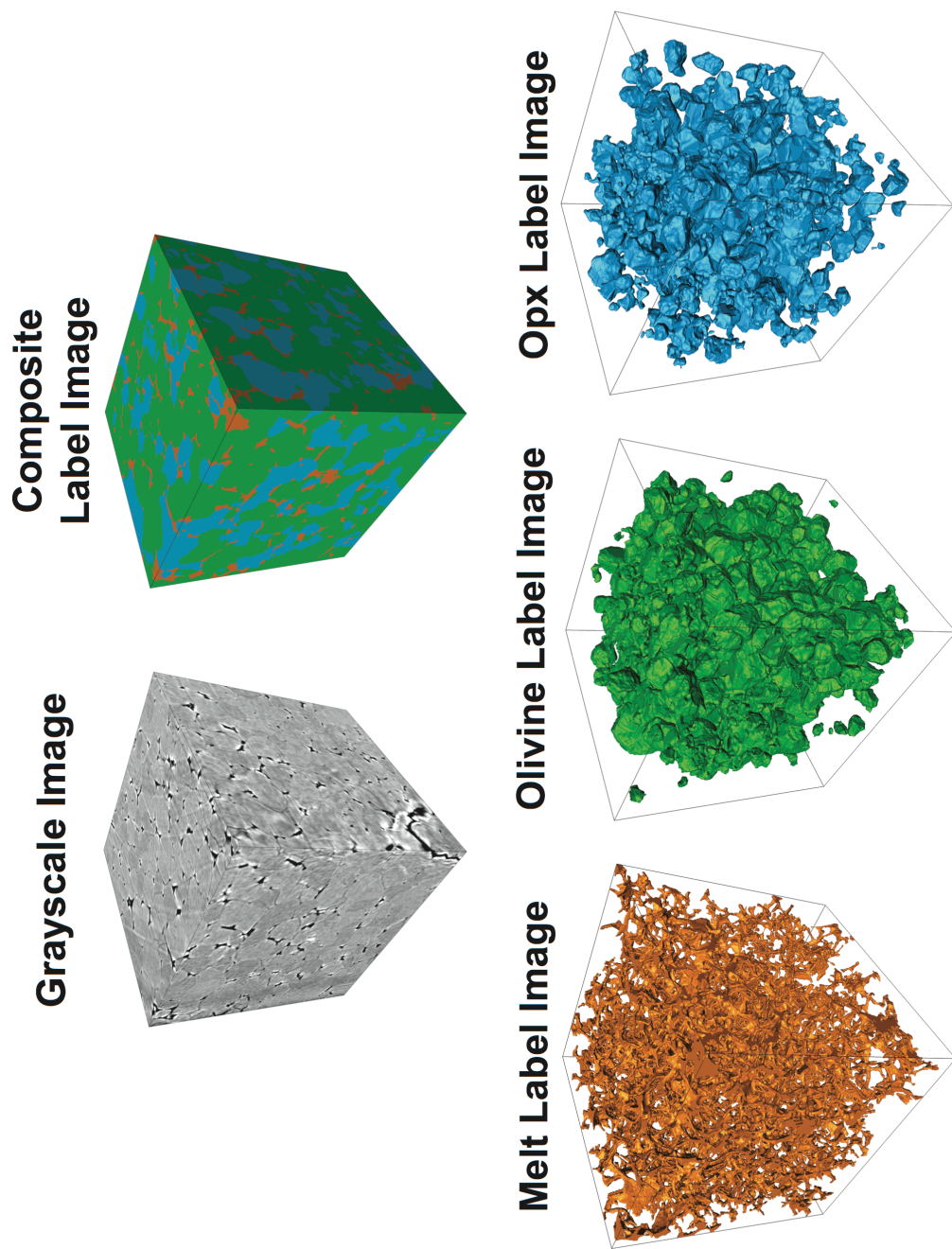


Figure 4.7: Illustrated workflow of data reduction technique. From left to right is original grayscale subvolume that was cropped from the sample and segmented label image. Label image can be decomposed into separate images of the melt, olivine, and opx. Border grains are removed from olivine and opx images prior to local melt fraction and grain size analysis.

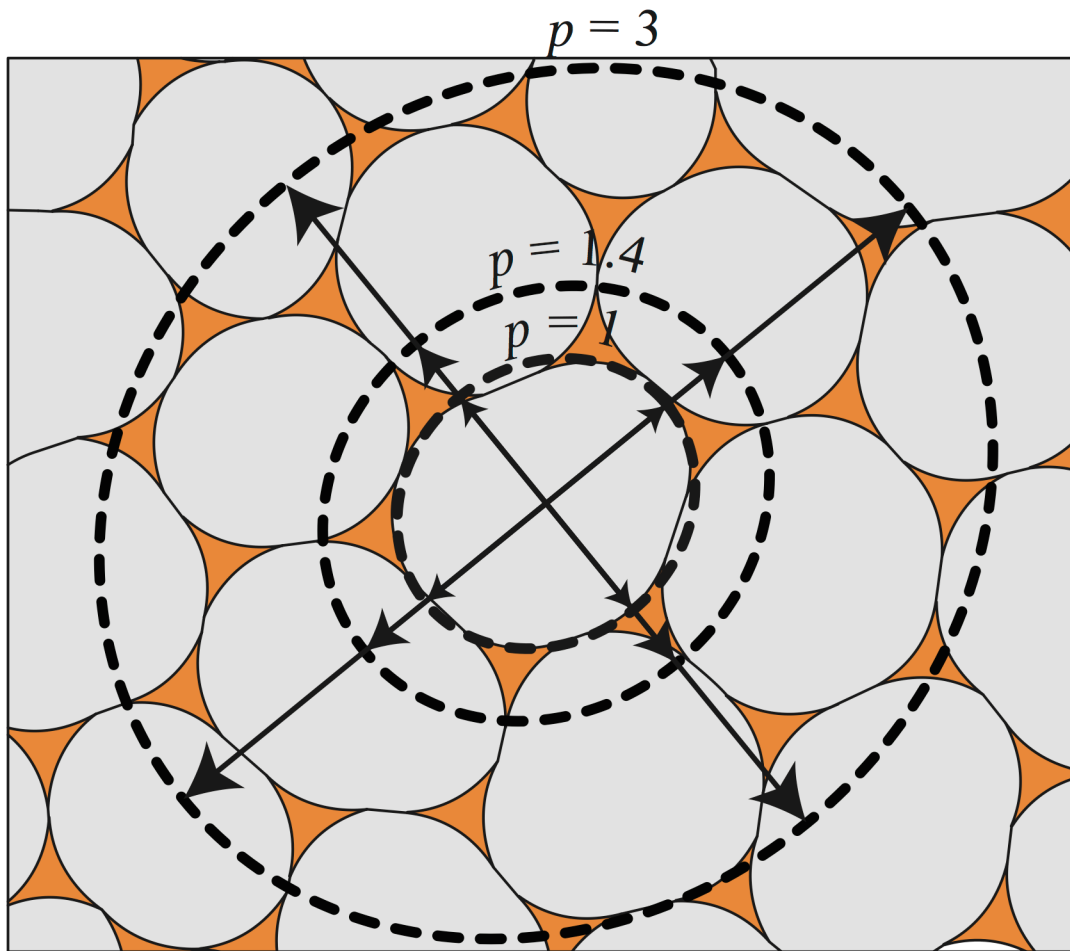


Figure 4.8 Schematic diagram of the local melt fraction analysis technique. Gray and orange represent grain and melt. Dotted lines are ellipsoid envelopes for different growth parameters $p = 1, 1.4$, and 3 . $p = 1$ ellipsoid is too small and miss melt. $p = 3$ ellipsoid is too large and counts melt from other grains. $p = 1.4$ is approximately the optimal value that enclose the melt immediately surrounding the grain.

adopted a dynamic measuring technique, in which a local melt fraction is measured for each grain and then plotted in a distribution. Each local melt fraction measurement was performed by counting melt voxels contained within an ellipsoidal envelope surrounding each grain. First, an ellipsoid was fitted to each grain (Fig. 4.8) using `ellipsoid_fit.m`, which is a freely available software on Matlab Central and is also included in Appendix C. The principle lengths and orientations of the ellipsoid are eigenvalues and eigenvectors, respectively, of the ellipsoid fit parameters. Next, we dilated the fitted ellipsoid by multiplication with a growth parameter p . Phase proportions were calculated by voxel counting within each ellipsoidal envelope. We looped through all grains in each subvolume and plotted them as a distribution.

Clearly, local melt fractions depend on p , so we calibrated our algorithm by computing the local melt fraction distributions for various values of p (Fig. 4.8). Values for p ranged from 1 (original fit to grain) to 4 (includes many grains). We wanted an ellipsoid envelop that enclosed only melt adjacent to each grain, which occurs for values of $p = 1$ to 1.4.

4.2.6 Characterizing grain size distributions

The grain size distribution of each subvolume was determined by estimating the equivalent diameter, which is defined as the diameter of a sphere having the equivalent volume as the grain, of each grain. First, an opening filter having a “ball-shaped” kernel was applied to the segmented grain label images. Second, a morphological watershed algorithm was used to approximate the solid-solid boundaries. The equivalent diameter was then measured for each grain.

The morphological watershed transform is completely automatic; however, there is a caveat: it sometimes incorrectly approximates grain boundaries. If grain boundaries are mostly melt-free, the morphological watershed transform can count multiple grains as a single grain. Aside from manually drawing grain boundaries, we do not have a method to correct for erroneous grain boundaries.

4.3 Results

4.3.1 Visual inspection of melt distribution

A visual inspection of a clump of opx grains (Fig. 4.9) near the bottom of hzb-14 ($\phi_n = 0.20$) qualitatively demonstrates lithological melt partitioning in a sample composed of olivine, opx, and basaltic melt. Mineral clumping occurs in higher frequency near the bottom of the sample where the melt fraction is much lower ($\phi \sim 0.04$) than the top. As pointed out in the figure – and holds true across all samples – olivine-rich regions are abundant sources of melt with respect to the opx-regions, which are nearly melt free for low melt fraction.

The presence of a reduced melt fraction that spans several or more grains has important implications for transport properties of the upper mantle. If present in the upper mantle, melt rich – or olivine rich because of lithologic melt partitioning – conduits may increase melt transport efficiency.

In our samples, olivine- and opx-rich regions are mixed more or less homogeneously in the sample and do not extend through the entire sample. Therefore, olivine-rich and opx-rich regions cannot conduct fluid flow in parallel. In the mantle, however, the reactive-infiltration instability is thought to juxtapose olivine with

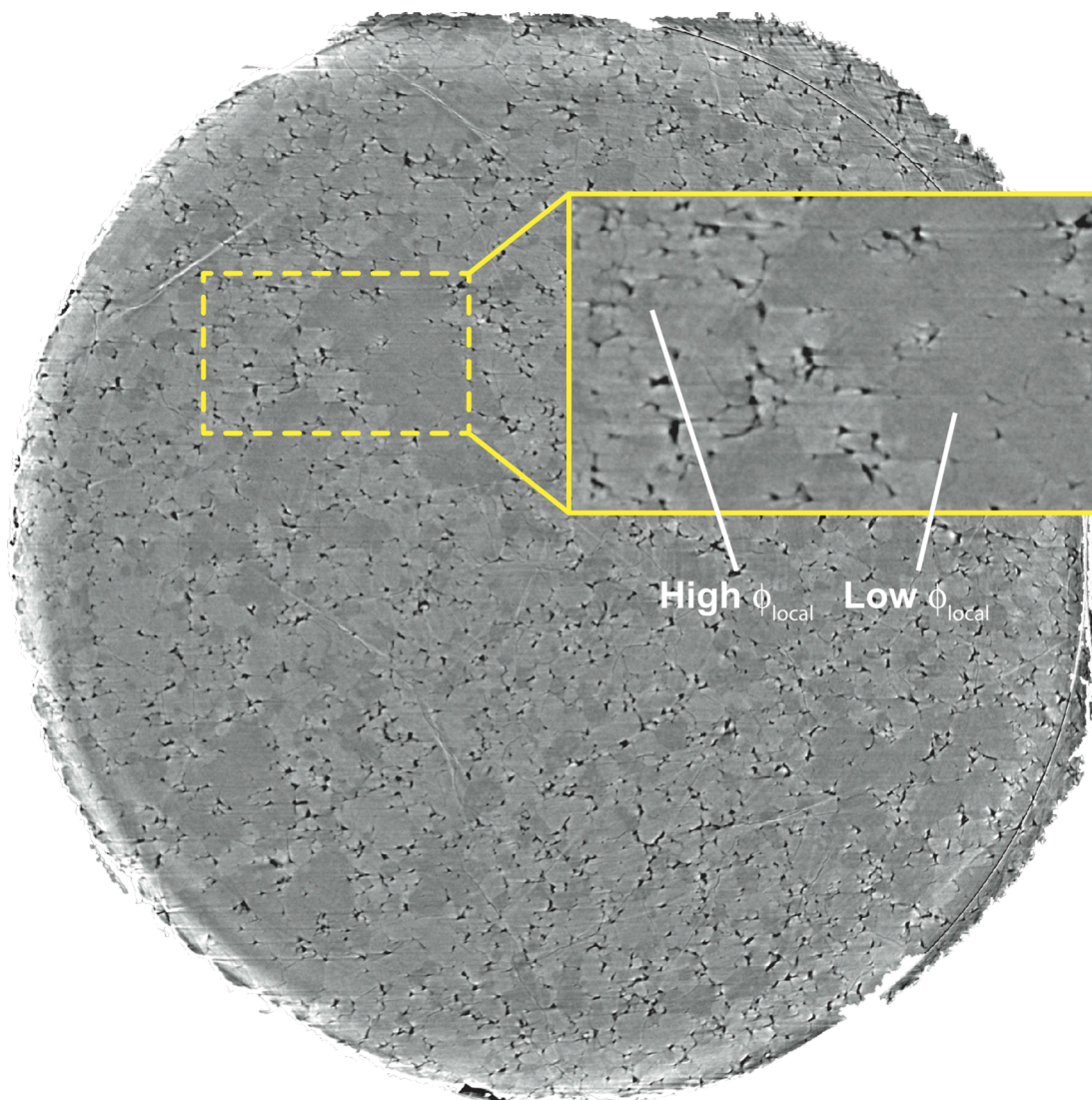


Figure 4.9: Visual inspection of tomography slice that contains ~4 vol. % melt. Effects of lithologic melt partitioning are clearly visible in the inset outlined in yellow: olivine-rich regions have considerably more melt than opx rich regions.

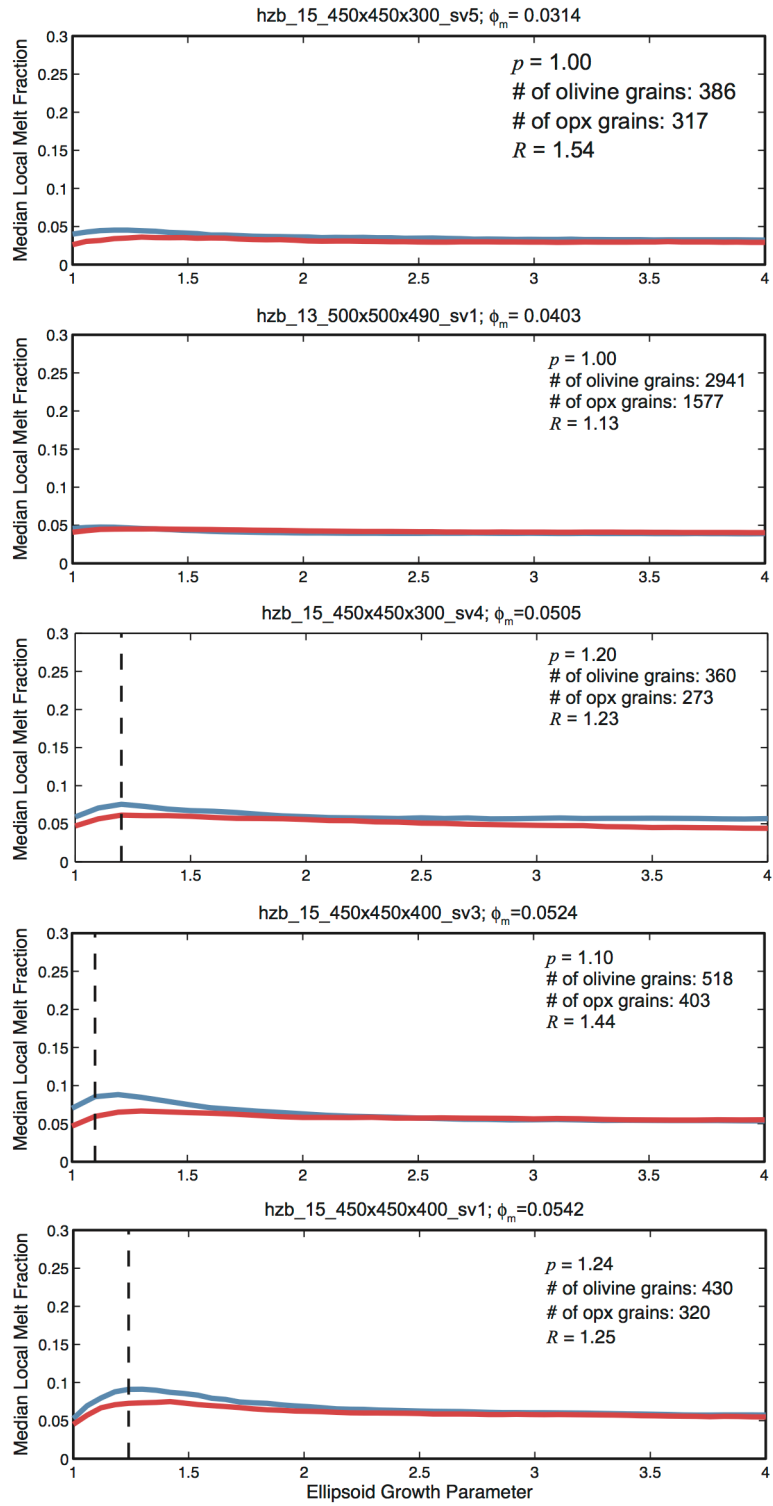
harzburgite (Kelemen et al., 1995a), which may allow the high melt fraction, high permeability olivine-rich region to transport melt in parallel with the low melt fraction, low permeability opx-rich region and increase the transport efficacy of the mantle.

4.3.2 Local melt fraction distributions

Local melt fraction distributions were computed for each subvolume. Use of p as a scaling factor for the ellipsoidal envelope assumes that the size of melt features scales with grain size. The difference between the median local melt fraction for olivine and opx are plotted as a function of growth parameter p (Fig. 4.10). As expected, the local melt fraction tends to the total measured melt fraction of the subvolume for very large values of p .

We report the minimum energy melt fraction for olivine and opx grains at p_{optimal} , which is the value of p that maximizes the difference between the median local melt fractions. We can see from Fig. 4.10, that the maximum difference in the median local melt fractions occurs between $p_{\text{optimal}} = 1.0$ and 1.4. For values less than p_{optimal} , voxels associated with adjacent melt features are missed. For p_{optimal} , neighboring grains dilute the measured local melt fraction.

We quantify the degree of melt partitioning by a parameter R , which is the median olivine local melt fraction divided by the median opx local melt fraction. We find that for all subvolumes, $R > 1$. Therefore, there is a higher local melt fraction associated with olivine grains than with opx grains. The difference between the median olivine and opx local melt fraction appears to increase with increasing melt



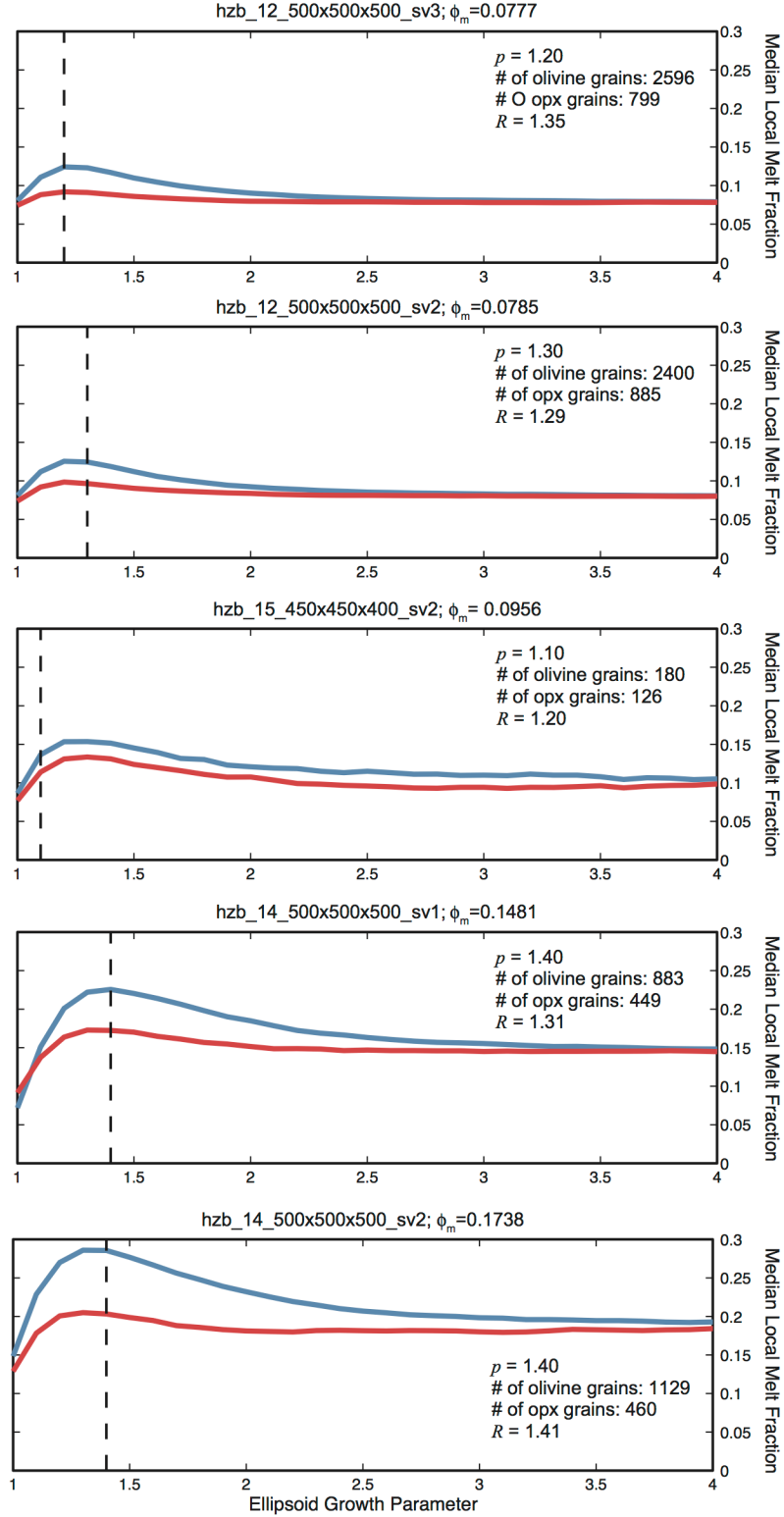


Figure 4.10: Median olivine and opx local melt fractions of each subvolume plotted as a function of growth parameter p . Optimal p values (plotted as dashed lines) are those that maximize the difference between olivine and opx median local melt fractions. We report the number of grains taken into account and the melt partition ratio at the optimal value of p .

fraction. Results are summarized in Table 4.2.

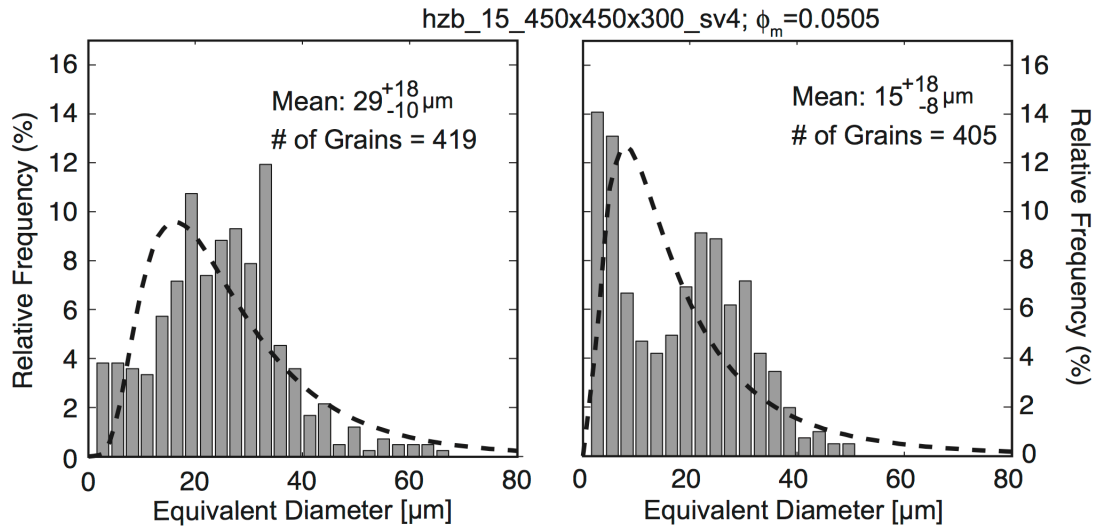
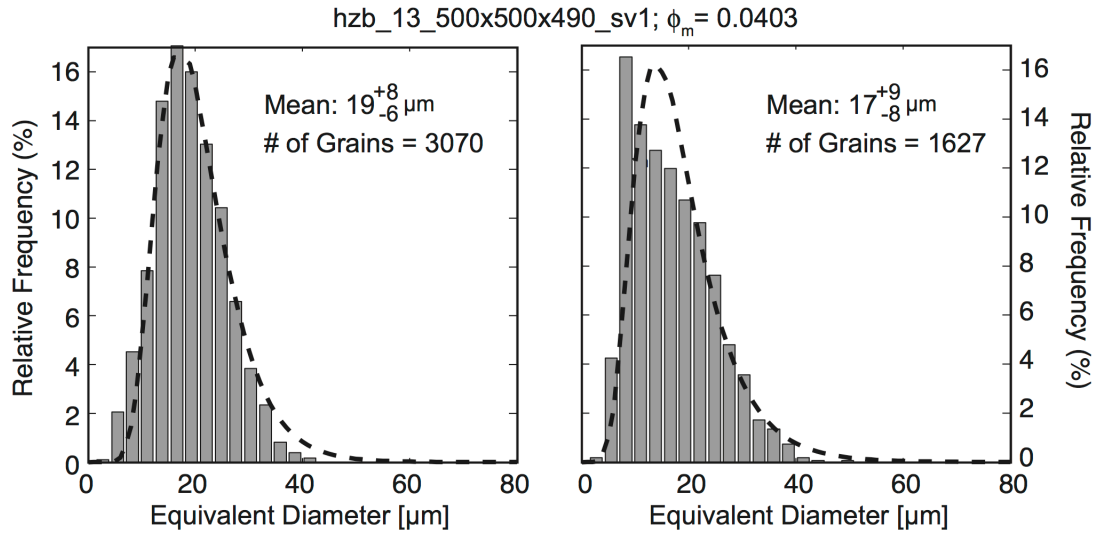
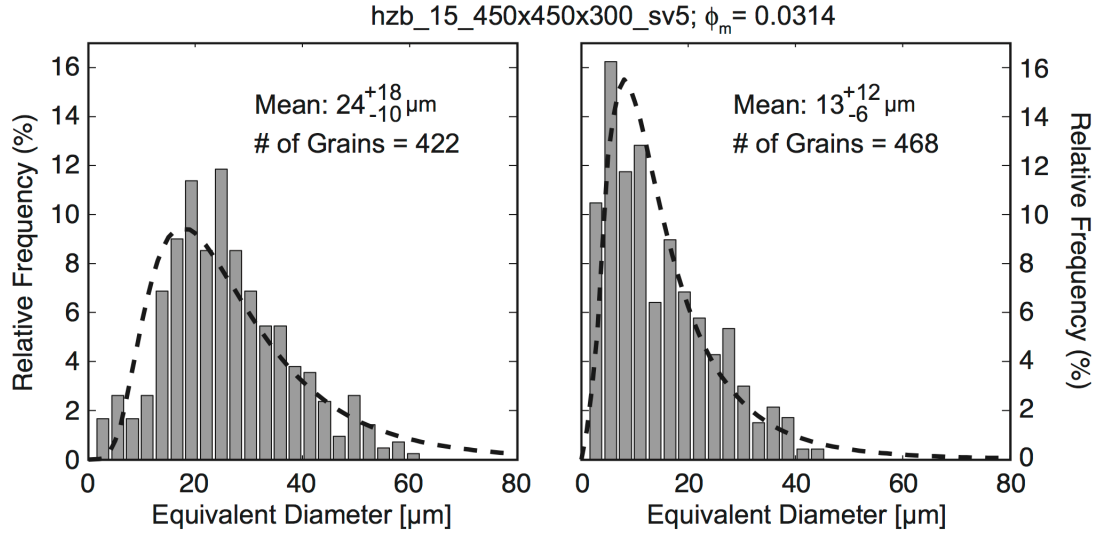
4.3.3 Grain size distributions

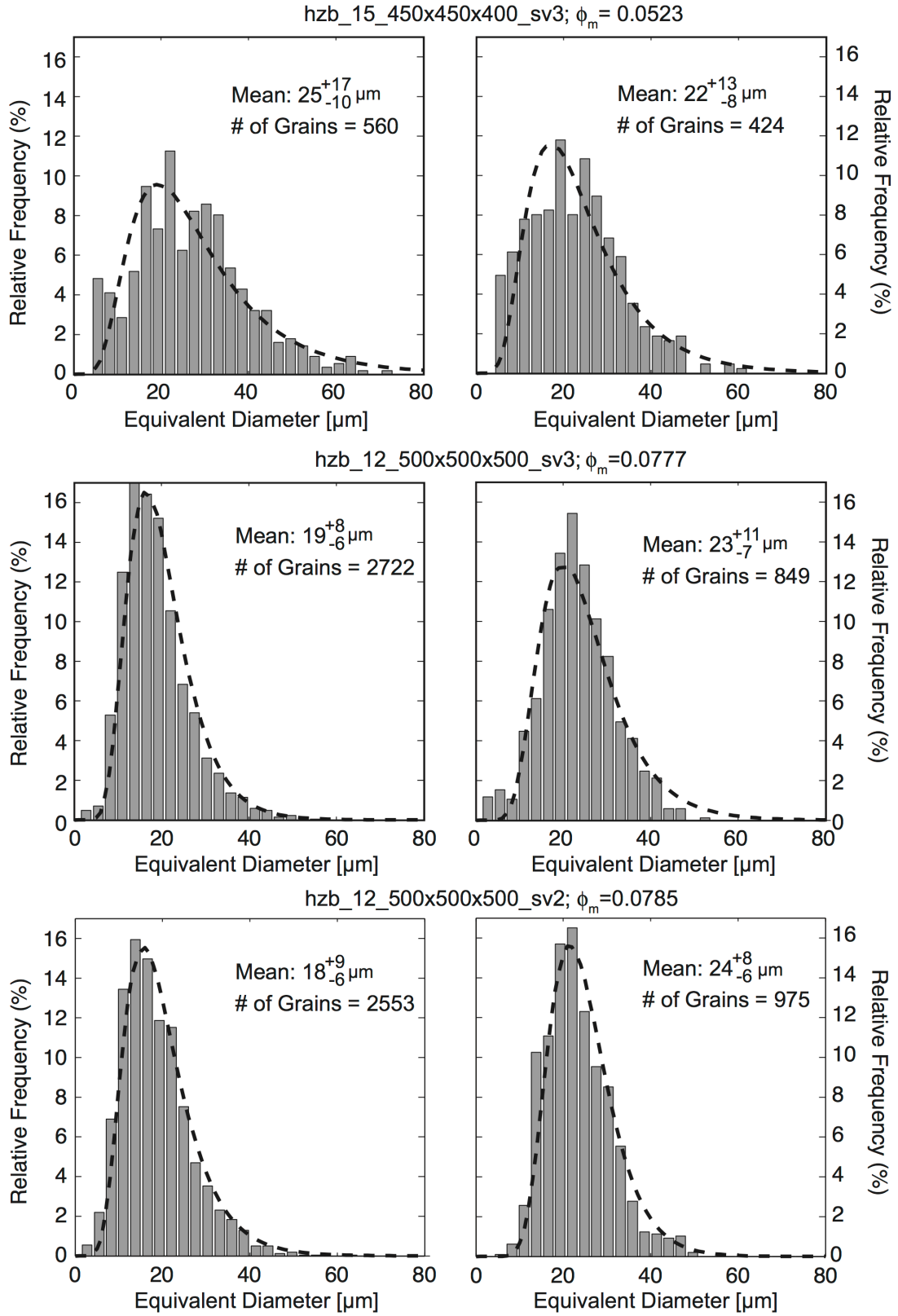
We compute equivalent diameter distributions for olivine and opx (Fig. 4.11). Equivalent diameter data appear to follow lognormal distributions. Correspondingly, we report the geometric mean and standard deviation as the mean grain size and width of grain size distribution. As expected, subvolumes containing order of 1000 grains have narrower distributions. As noted in Miller et al. (2014), the automated watershed transform that was used to separate 3-D grain data produces a more accurate grain size distribution when the melt fraction is higher, since grain boundaries are more easily distinguished if they are coated by melt. Melt-free triple junctions and dry grain-grain boundaries occur with increasing frequency as the melt fraction decreases.

In order to understand the kinetics of grain growth in our polymineralic aggregate and to evaluate the efficiency of grain growth via chemical diffusion through the interconnected melt network, we plot mean grain size of olivine and opx as a function of melt fraction (Fig. 4.12). Though there is significant overlap of the grain size distributions, the median opx grain size increases systematically with melt fraction while the median olivine grain size is insensitive to changes in melt fraction. Interestingly we do not see evidence of grain pinning in the olivine grain size data.

4.4 Discussion

4.4.1 Melt concentration due to lithologic melt partitioning





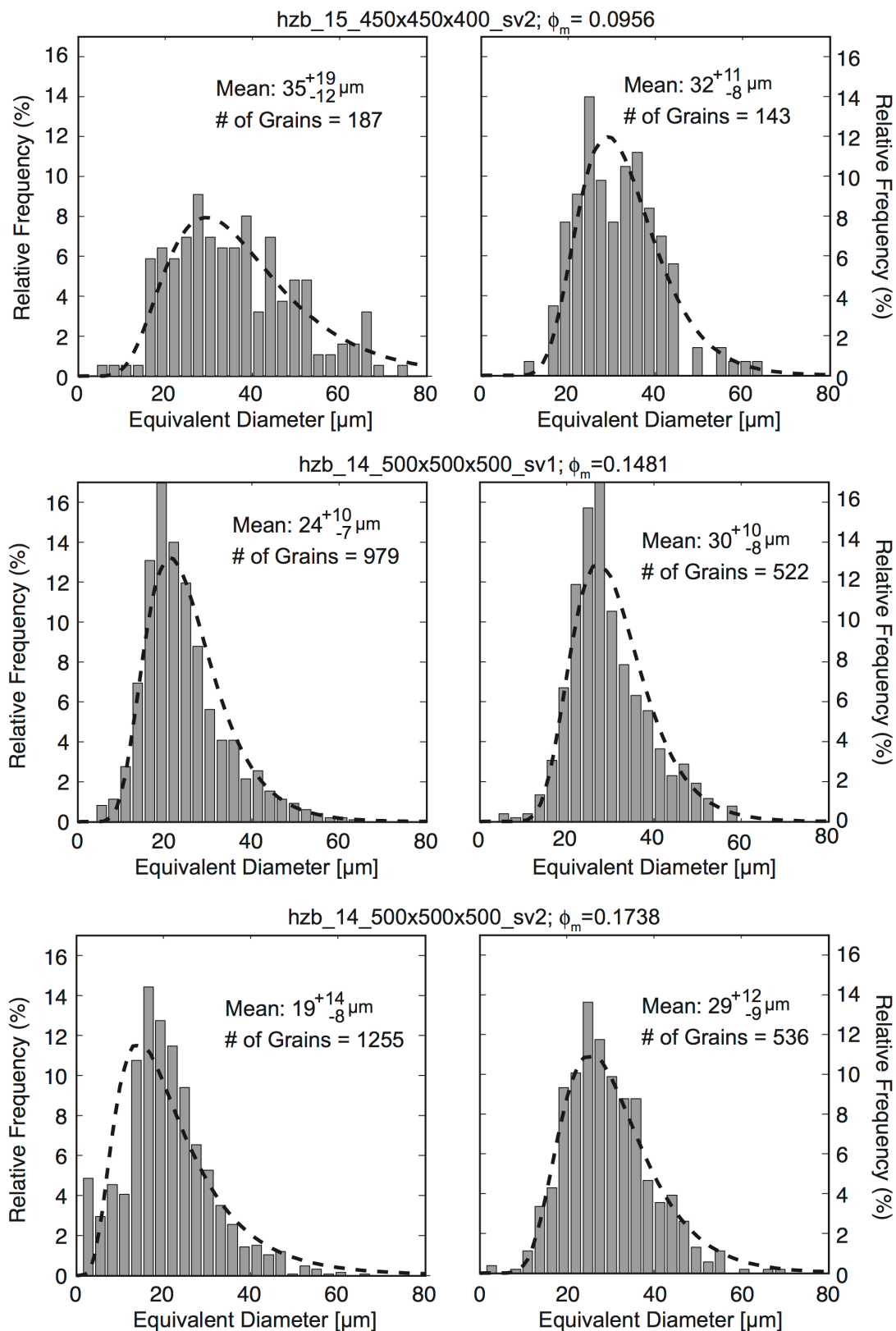


Figure 4.11: Olivine (left) and opx (right) equivalent diameter distributions measured for each subvolume. Mean and standard deviations are obtained by computing the geometric mean and (1σ) standard deviations. Number of grains used in statistics is also reported.

Subvolume	Sintering Duration [hrs.]	Nominal Melt Fraction	$\phi_{\text{melt}}^{\text{global}}$	$\phi_{\text{olivine}}^{\text{global}}$	$\phi_{\text{opx}}^{\text{global}}$	$\phi_{\text{olivine}}^{\text{global}} / \phi_{\text{opx}}^{\text{global}}$	$\bar{d} [\mu\text{m}^2]$	R
hzb_13_60mm_500x500x490_sv1	336	0.02	0.0403	0.673	0.2867	2.3474	18	1.13
hzb_12_60mm_500x500x500_sv2	200	0.10	0.0785	0.5756	0.3459	1.6641	20	1.29
hzb_12_60mm_500x500x500_sv3	200	0.10	0.0770	0.6122	0.3108	1.9698	20	1.35
hzb_14_60mm_500x500x500_sv1	200	0.20	0.1481	0.4876	0.3643	1.3385	26	1.31
hzb_14_60mm_500x500x500_sv2	200	0.20	0.1738	0.4482	0.3780	1.1857	22	1.41
hzb_15_50mm_450x450x400_sv2	168	0.05	0.0956	0.6714	0.2330	2.8815	34	1.2
hzb_15_60mm_450x450x450_sv3	168	0.05	0.0524	0.6680	0.2796	2.3891	23	1.44
hzb_15_60mm_450x450x300_sv4	168	0.05	0.0505	0.6962	0.2533	2.7485	18	1.23
hzb_15_60mm_450x450x300_sv5	168	0.05	0.0314	0.7878	0.1808	4.3573	17	1.54

Table 4.2: Summary of harzburgite results. Numerical values in the subvolume ID represent the sample name, sample-to-detector distance, subvolume dimensions (measured in voxels), and subvolume identification number, respectively. The superscripts “global” refer to the measured phase volume fractions of each subvolume. \bar{d} is the geometric mean equivalent diameter. R is the melt partition ratio.

Our results are strong evidence that spatial variations in mineralogy cause lithologic melt partitioning in partially molten harzburgite. However, the length scale over which spatial gradients in surface energy can segregate melt is not currently constrained. If the effect of lithologic melt partitioning is short-range, i.e. can only cause melt fraction heterogeneity in the immediate proximity of low-surface energy grain surfaces, the permeability structure of the upper mantle should adhere closely to the mineralogical structure of the geological formation. However, if the range of lithologic melt partitioning reaches beyond the proximity of adjacent grains, it may enhance the melt transport capabilities of the upper mantle.

The sharp contrast in melt fraction in close proximity to opx-rich regions suggests that lithologic melt partitioning is short-range. However, in a closed system with a finite melt fraction, conservation of mass necessitates that even a tiny enhancement of the local melt fraction be compensated by a decrease in melt fraction elsewhere in the sample.

4.4.2 Lithologic melt partitioning and transport properties

Lithologic melt partitioning has the potential to enhance the permeability of partially molten harzburgite. For a monomineralic system, permeability depends only on the spatial distribution of melt in the volume. The presence of a low wettability mineral phase will perturb the otherwise uniform melt distribution, in which case the effective permeability is some complicated mixing between the permeability of two end-member mineralogies. Unfortunately, determining the actual mixing relation for harzburgite effective permeability requires computing permeabilities of both partially

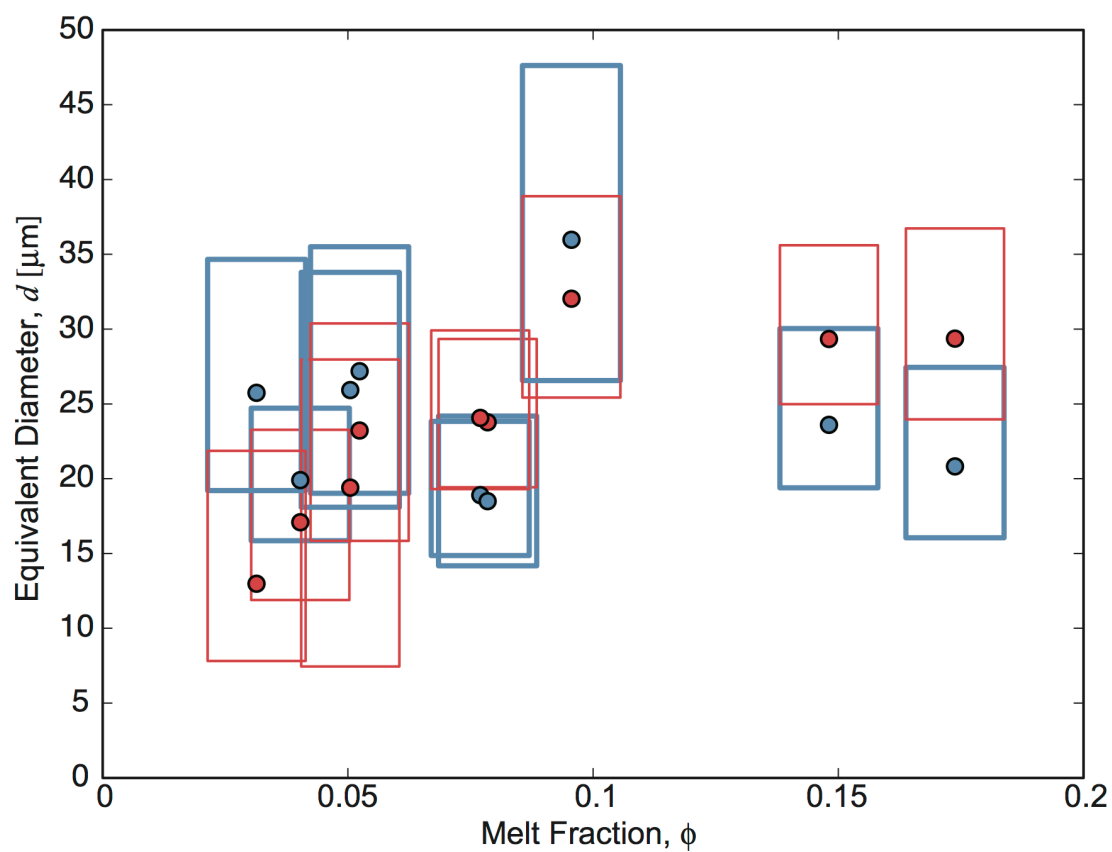


Figure 4.12: Equivalent diameter variation as a function of melt fraction. Blue and red represent olivine and opx, respectively. Circles are the median equivalent diameters. Bottom and top of boxes are the 25% and 75% quartiles, respectively.

molten pure olivine-basaltic melts and opx-basaltic melts for various melt fractions, which we do not have. Nevertheless, percolation theory suggests the effective permeability of a homogeneously mixed olivine-opx aggregate approaches the geometric mean of the individual partially molten dunite and pyroxenite end-members (Madden, 1976). However, if olivine and opx-rich regions are for some reason organized into conduits, the two regions will conduct fluid flow in parallel, and the olivine-rich regions will dominate fluid flow. Conversely, if olivine- and opx-rich regions are overlaid as layers that are oriented perpendicular to the flow direction, the lower permeability region will determine the effective permeability.

4.4.3 Geological implications for lithologic melt partitioning

There is no evidence that lithologic melt partitioning can create a mineralogical heterogeneity; an initial mineralogical heterogeneity needs to be present. The reaction infiltration instability (RII) is a good candidate for establishing an initial mineralogical heterogeneity. The RII is a positive feedback processes in which dissolution of opx in a harzburgitic mantle by a melt that is undersaturated with respect to opx leads to an increase in melt flux that further promotes opx dissolution (Daines and Kohlstedt, 1994; Kelemen et al., 1997, 1995a). Numerical modeling using multiphase flow theory has shown that the RII is capable of forming high melt fraction dunite conduits whose thicknesses range from tens to thousands of meters (Aharonov et al., 1995; Kelemen et al., 1995a; Spiegelman et al., 2001). More recently, the RII has been confirmed to produce high melt fraction dunite conduits in laboratory experiments (Pec et al., 2015). If these dunite conduits are present in the

upper mantle, they may constitute a thermodynamic gradient that further segregates melt in the upper mantle.

Lithologic melt partitioning may help to stabilize the formation of high-melt fraction conduits that form as a result of the RII. Spiegelman et al. (2001) suggests that once the opx supply has been depleted, the melt fraction will continue to eat away at the side of the conduits so as to replenish the melt fraction in the conduits lost to buoyancy. This is an unstable process that causes opx dissolution to progress until olivine is the sole mineral component of the upper mantle. However, field observations of banded dunite-harzburgite formations in the Oman ophiolite (Kelemen et al., 1995a) suggest that dunite conduits are persistent features of the upper mantle if we assume a steady-state mid-ocean system. Therefore, an additional mechanism is required to sustain high melt fraction in the dunite conduits. The observed lithologic melt partitioning in our harzburgite samples may provide a mechanism for driving melt into the dunite channels, replenishing the melt supply in the high-melt fraction dunites.

4.4.4 Grain size and melt fraction

We attribute the increase in mean opx grain size with melt fraction (Fig. 4.12) and the insensitivity of olivine to melt fraction to differences in wetting properties of the mineral components. Chemical diffusion through an interconnected melt network is a more efficient means of growing grains than grain boundary diffusion (Watson, 1999). If the dihedral angle associated with a phase boundary is greater than 60° , a threshold melt fraction is required to maintain interconnectivity of the melt network;

otherwise melt forms isolated pockets at grain corners. For this scenario, grain boundary diffusion is the sole mode of transport for the material required to grow grains. Conversely, for high melt fraction, melt forms an interconnected network in the presence of both olivine and opx. As the melt fraction decreases, the melt network begins to lose connectivity around opx grains, disconnecting them from their chemical supply.

There is evidence of a tradeoff between melt-assisted diffusion and grain-boundary diffusion in our samples (Fig. 4.12). Below the percolation threshold, opx grains grow via grain-boundary diffusion. Olivine grains, however, which form a dihedral angle of $\sim 35^\circ$ (Waff and Bulau, 1982) with basaltic melt, will maintain contact with the melt network at all melt fractions. Therefore, olivine grain growth should be relatively insensitive to melt fraction.

4.5 Conclusion

We used high-resolution X-ray μ -CT to image the 3-D microstructure of partially molten harzburgites that contain a range of melt fractions. A novel methodology was applied to resolve the density contrast at olivine-basalt, opx-basalt, and olivine-opx interfaces. We computed local melt fraction distributions for olivine and opx grains by fitting ellipsoidal envelopes to each grain. We found that melt partitions in about a 1.1 to 1.5 ratio between olivine and opx for total nominal melt fractions 0.02 to 0.20, which we attribute to spatial variations in surface energy associated with low surface energy density olivine interfaces and high surface energy density opx interfaces. The measurable melt partitioning in harzburgitic systems

warrants a microstructural evaluation of transport properties permeability and electrical conductivity as well as numerical modeling of larger magmatic systems composed of substantial proportions of olivine and opx.

Chapter 5: Permeability and electrical conductivity of partially molten harzburgite

Abstract

Modeling melt transport and correctly interpreting electromagnetic data of the upper mantle beneath mid-ocean ridges require robust, microstructure-based constraints on the constitutive equations that relate permeability and electrical conductivity to melt fraction, respectively. Differences in the wetting properties of minerals are thought to alter transport properties of partially molten mantle rock. The presence of orthopyroxene, for example, is thought to decrease the connectivity of the melt network if the local melt fraction dips below the melt fraction required for maintaining an interconnected network. Since opx is a primary constituent of the upper mantle, any relation between transport properties and melt fraction must consider its effects.

We examined the effect of opx on the permeability and electrical conductivity of partially molten rock aggregates composed of olivine, opx, and basaltic melt over a range of nominal melt fractions ($\phi_n = 0.02$ to 0.20). Synthetic olivine-opx-melt samples were prepared by isostatically hot-pressing powdered mixtures of oxides and natural, high-alumina basalt at 1.5 GPa and 1350 °C for a minimum of one week. Experimental charges were cored and imaged using synchrotron-based X-ray micro-computed tomography. The resulting 3-D images constitute digital rock samples, on which numerical laminar flow and direct current simulations were conducted. Permeabilities and electrical conductivities of olivine-opx-melt samples were

compared to those composed of pure olivine and basaltic melt at similar melt fractions. We found that all olivine-opx-melt permeability data plot along the permeability-melt fraction curve for olivine-melt if we compensate for intersample variations in the mean grain size. Interestingly, we found that the bulk electrical conductivity of harzburgite is systematically lower than that of dunite.

5.1 Introduction

The capacity of the upper mantle to transport melt at mid-ocean ridges and conduct electricity largely depends on the interconnectivity of the grain-scale melt network. For a dihedral angle less than 60° , melt forms an interconnected network at any melt fraction; otherwise a threshold melt fraction is required to maintain melt interconnectivity. Since olivine forms a dihedral angle of $\sim 35^\circ$ (Fig. 5.1) with basaltic melt (Waff and Bulau, 1982), melt transport in the upper mantle, which is primarily composed of olivine, is thought to be efficient. However, field observations suggest the mantle composition is closer to a harzburgite, containing as much as 40 vol. % orthopyroxene (opx), which forms a dihedral angle of $\sim 75^\circ$ with basaltic melt (Fig. 5.1) (Toramaru and Fujii, 1986). Therefore, if the threshold melt fraction needed for melt interconnectivity is not maintained everywhere, opx-rich regions will contain isolated melt, decreasing the permeability and electrical conductivity of the rock.

Though we know the permeability and electrical conductivity of mantle rock is some complicated average that depends on the modal proportion and spatial distribution of olivine and opx (Madden, 1976), the exact influence of opx on the transport properties of mantle rock is difficult to constrain using conventional rock

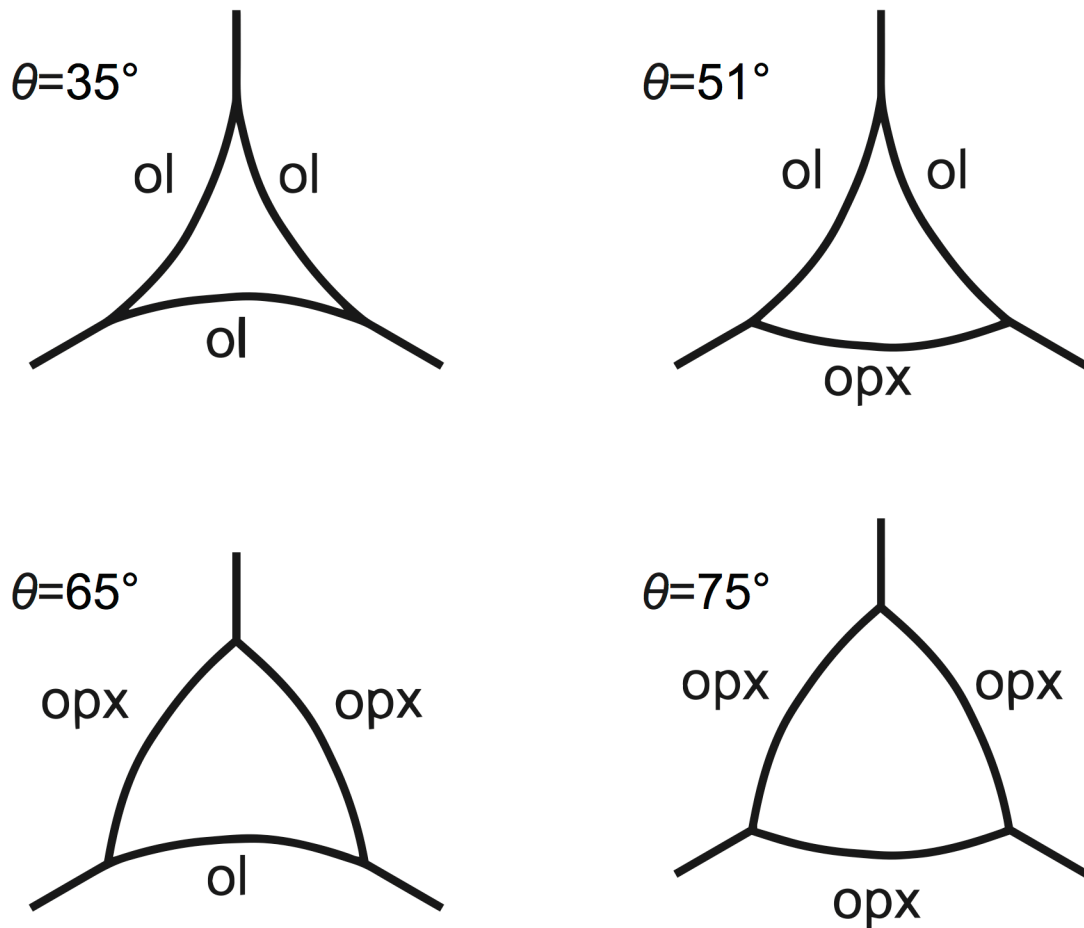


Figure 5.1: Comparison of olivine-olivine-olivine, opx-opx-opx, olivine-olivine-opx, and opx-opx-olivine triple junctions. Modified from Zhu and Hirth (2003). θ represents the effective dihedral angle of the triple junction. Melt in pure olivine aggregate will always form an interconnected network. Melt in pure opx aggregate will form isolated melt pockets unless a threshold melt fraction is attained. The presence of opx increases the effective dihedral angle of the triple junction.

physics experiments. Microscopy analysis of partially molten rocks composed of olivine, opx, and basaltic melt offer useful information regarding the connectivity of melt in polymineralic system. For example, Toramaru and Fujii (1986) analyzed the dihedral angle distributions of synthetic olivine-opx-melt samples. They found that the number of isolated melt pockets and melt-free triple junctions increases with increasing opx proportion. They attributed their result to the tendency for melt to form isolated melt pockets when adjacent to high surface energy density phase boundaries (e.g. opx-melt interfaces). Isolated melt pockets do not facilitate melt transport and contribute only minorly to electrical conductivity of the aggregate.

The influence of high dihedral angle associated with opx-bearing triple junctions on permeability was assessed using network permeability models (Zhu and Hirth, 2003). Assuming melt formed an interconnected network only along triple junctions, Zhu and Hirth (2003) computed permeabilities for three-phase systems containing various proportions of olivine, opx, and interstitial basaltic melt. Despite the ability of opx to reduce melt interconnectivity, they found that a system composed of 40 vol. % (proportion of opx volume to grain volume) only reduced permeability by a factor of ~ 2 relative to an olivine-melt system at melt fraction of 0.01. As the number of wetted triple junctions required to maintain an interconnected network approached the percolation threshold (39% triple junctions are wetted) permeability drops off rapidly with melt fraction: at melt fraction 0.01, 60 vol. % opx results in over four orders of magnitude reduction in permeability. Network models by Zhu and Hirth (2003) provide strong motivation to examine synthetic systems composed of olivine, opx, and basaltic melt.

An additional influence of opx on the grain-scale distribution of melt – and potentially the transport properties – is the tendency of melt to localize around low-energy interfaces. This phenomenon known as lithologic melt partitioning (Jurewicz and Watson, 1985; Watson, 1999), has been verified in variety analogue systems (e.g. quartz-clinopyroxene, calcite-fluorite, and quartz-fluorite) (Watson, 1999) and recently in Chapter 4 of this manuscript for partially molten rocks composed of olivine and opx. Since transport properties depend on melt fraction, lithologic melt partitioning may affect the permeability and electrical conductivity on an aggregate scale, and if coupled with an additional mechanism that forms mineralogical heterogeneity larger than the grain-scale, lithologic melt partitioning may drastically modify the efficiency of melt transport in the mantle.

As a first step to understanding how mineralogical heterogeneity affects melt transport in the upper mantle, we seek to quantify the grain-scale permeability and electrical conductivity of partially molten harzburgite as a function of melt fraction. Since permeability and electrical conductivity are technically challenging to measure experimentally, we adopt a digital rock physics approach. We synthesize partially molten harzburgites that have various proportions of basalt and a constant olivine to opx volume ratio. High-resolution, three-dimensional images were taken using synchrotron-based micro-computed tomography. Virtual fluid flow and direct current experiments were conducted using the melt geometries to compute permeability and electrical conductivity. Permeabilities and electrical conductivities of partially molten harzburgite samples were compared to those computed for olivine (Chapter 2 and 3 of this manuscript).

5.2 Methods

5.2.1 Sample preparation of harzburgite samples

Harzburgite samples were prepared by hot, isostatic pressing of oxide-basalt mixtures in piston-cylinder apparatuses at 1350 °C and 1.5 GPa. The composition of the primary oxide mixture was based on the chemical composition of a natural Southwest Indian Ridge harzburgite (Dick, 1989) and adjusted for each melt fraction so that we achieved a nominally constant 3 to 2 (olivine to opx) volumetric ratio and melt fraction 0.02, 0.05, 0.10 and 0.20 after sintering. Details of the sample preparation procedure are discussed in Section 4.2.1. The oxide mixture was homogenized over ethanol by six six-hour homogenization cycles in an automatic agate mortar and pestle. Pulverized natural basalt was added in various proportions to the oxide mixture to attain total nominal melt fractions of 0.02, 0.05, 0.10, and 0.20 under run conditions. Each oxide-basalt mixture was homogenization using the same procedure as the primary oxide mixture.

Upon completion of the experimental runs, experimental charges were quenched by turning off the power while maintaining a steady flow of cold water around the pressure vessel. 1 mm cylindrical cores were drilled from each sample along the cylindrical sample axis.

5.2.2 Imaging procedure

Following Zhu et al. (2011), cylindrical harzburgite samples were imaged using a 24.4 keV synchrotron light source at 2BM of the Advanced Photon Source,

Argonne National Laboratory, Argonne, IL. Image reconstruction was performed using the software package Tomopy (Gürsoy et al., 2014). Refer to Section 4.2.2 for a detailed description of the imaging procedure.

5.2.3 Subvolume Selection

Several smaller, computationally manageable data subsets, which we call “subvolumes,” were cropped from each reconstructed digital sample. Subvolume sizes and locations were selected so as to avoid long-wavelength variations in the measured melt fraction and decompression fractures. Wherever possible, we sought subvolume sizes as large as we could computationally handle ($500 \times 500 \times 500$ voxels³ for permeability computations and $400 \times 400 \times 400$ voxel³ for electrical conductivity computations). If decompression fractures or the vertical melt anomaly prevented us from selecting such a larger subvolume, we opted for a smaller subvolume; though even the smallest subvolume contains greater than 300 grains.

5.2.4 Image segmentation

Avizo[®] was used to perform image segmentation. In order to characterize the melt distribution and transport properties, each grayscale subvolume needed to be converted to a label image, where each grayscale voxel was assigned a value of 1, 2, or 3 for basaltic glass, olivine, or opx, respectively. We developed a semi-automatic segmentation workflow that we applied to all subvolumes. First, melt was segmented using a combination of Avizo’s local thresholding module and tophat global threshold (Vincent, 1993). Thin decompression fractures were manually removed from the

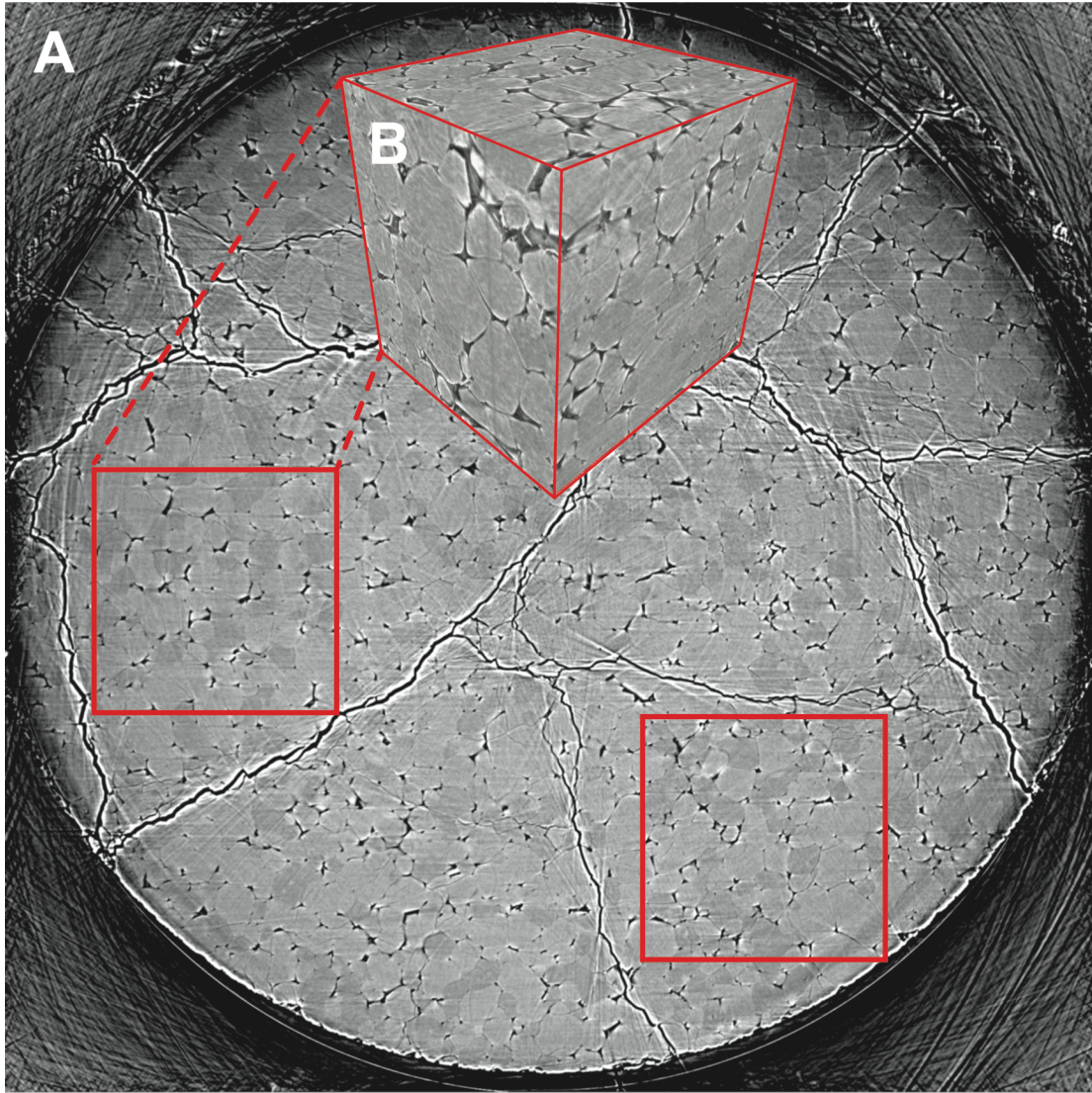


Figure 5.2: Subvolume selection for harzburgite suite. (A) Reconstructed tomography slice from harzburgite sample containing ~ 5 vol. % basaltic glass (quenched melt). Clearly visible is olivine (light granular phase), opx (dark granular phase), and basaltic glass (dark, interstitial phase). Linear features are decompression fractures. White streaks are image artifacts from the reconstruction algorithm. Boxes outlined by red denote locations where subvolumes were cropped. (B) 3-D visualization of harzburgite subvolume.

image by overlapping the trial segmentation with a morphological erosion and dilation of the image, applied sequentially using a $2 \times 2 \times 2$ voxel³ ball-shaped kernel.

Subtle contrast and bright imaging artifacts at the grain edges prevented us from applying the same local threshold technique to differentiate the opx from the olivine. We used Avizo's morphological watershed transformation (Beucher and Meyer, 1992) to separate grains and then handpicked opx grains from the aggregate. Grains that were incorrectly separated were corrected using a propagating contour tool. Once all of the opx grains were differentiated from olivine, the watershed basins were removed by simultaneously dilating the olivine and opx grain images. The resulting image was smoothed using an isotropic Gaussian kernel to remove the jaggedness imposed by the morphological watershed transform. The segmented melt image was superposed on the olivine-opx label image. The result was a very accurate segmentation of the melt and a slightly less accurate approximation of the olivine-opx grain boundaries.

5.2.5 Computation of permeability and electrical conductivity

Permeabilities of our partially molten harzburgite subvolumes were obtained using Avizo's XLab Hydro Absolute Permeability Experiment Simulation module, which mimics an actual permeability measurement. The melt geometry was discretized according to the original voxel spacing (1 voxel = 0.7 μm). Velocity and pressure fields were obtained by solving the Stokes Equations using the artificial compressibility method (Chorin, 1967) on a staggered finite-volume grid. Refer to Section 2.3.5 for a detailed description of the numerical model.

Permeability was obtained by applying Darcy's Law to the model output. The volume-averaged velocity field was used in place of the so-called Darcy velocity (Whitaker, 1998). Permeability is a function of only the melt geometry; external quantities, such as the imposed pressure gradient and viscosity, are divided out in the volume-average step and do not bear on permeability.

Bulk electrical conductivities of each subvolume were computed using Finite-Difference Electrical Conductivity Calculator (FDECC), which is a Matlab-based direct current experiment simulator that we built in-house. FDECC is based on the finite-difference formulation derived by Garboczi (1998). FDECC discretized the subvolume label image according to the original voxel spacing. Electrical conductivities were assigned to each voxel. We obtained the electrical potential scalar field by solving the current continuity equation (Laplace Equation) using the implicit finite-difference method. The volume-averaged current density was computed from the electric potential field. The bulk electrical conductivity of the label image was obtained by applying Ohm's Law to the model output. Refer to Section 3.2.4 for details about the direct current simulation.

5.2.6 Characterizing grain size distributions

In addition to melt fraction and melt interconnectivity, permeability depends on the grain size. Our subvolumes exhibit a significant variation in their mean grain sizes. In order to fairly evaluate the dependence of permeability on measured melt fraction, we divided each permeability value by the square of the geometric mean grain size measured for each subvolume.

Grain sizes distributions of each subvolume were determined by measuring the equivalent diameter, which is the diameter of a sphere having the equivalent volume as the grain. First, a generous opening filter having a “ball-shaped” kernel was applied to the segmented grain label images. Second, a morphological watershed algorithm was used to approximate the grain-grain boundaries. The equivalent diameter was measured for each grain.

The morphological watershed transform is completely automatic, so it is very useful for analyzing a large number of grains. However, there is a caveat: the morphological watershed transform often incorrectly approximates grain boundaries. If grain boundaries were mostly melt-free, the morphological watershed transform sometimes counted multiple grains as one grain.

5.3 Results

5.3.1 Statement about uncertainty

Melt fraction error bars in Fig. 5.3 and 5.4 do not reflect random, Gaussian error. Instead, the left and right end of each error bar is the measured melt fraction after a 1-pixel contraction and dilation of the 3-D melt geometry, respectively. Therefore, a meaningful comparison of the olivine-melt and olivine-opx-melt permeability and electrical conductivity datasets must be conducted on their corresponding minimum and maximum melt fractions. This method of using morphological contraction / dilation to define minimum and maximum error bars for measured phase proportions is rather crude, since it likely overestimates the effect blurring due to instrument error and discretization of the sample geometry; however

Subvolume	Nominal		Bulk			
	Melt Fraction	$\phi_{\text{melt}}^{\text{global}}$	$\phi_{\text{olivine}}^{\text{global}}$	$\phi_{\text{opx}}^{\text{global}}$	d [μm^2]	Permeability [m^2]
hzb_13_60mm_500x500x490_sv1	0.02	0.0403 (-0.0325/+0.0528)	0.6730	0.2867	18	8.33×10^{-16}
hzb_12_60mm_500x500x500_sv2	0.10	0.0785 (-0.0471/+0.0598)	0.5756	0.3459	20	5.56×10^{-15}
hzb_12_60mm_500x500x500_sv3	0.10	0.0770 (-0.0471/+0.0603)	0.6122	0.3108	20	4.13×10^{-15}
hzb_14_60mm_500x500x500_sv1	0.20	0.1481 (-0.0548/+0.0592)	0.4876	0.3643	26	4.36×10^{-14}
hzb_14_60mm_500x500x500_sv2	0.20	0.1738 (-0.0582/+0.0614)	0.4482	0.3780	22	7.18×10^{-14}
hzb_15_60mm_450x450x400_sv2	0.05	0.0956 (-0.0582/+0.0370)	0.6714	0.2330	34	1.81×10^{-14}
hzb_15_60mm_450x450x450_sv3	0.05	0.0524 (-0.0337/+0.0311)	0.6680	0.2796	23	4.45×10^{-15}
hzb_15_60mm_450x450x300_sv4	0.05	0.0505 (-0.0259/+0.0332)	0.6962	0.2533	18	1.99×10^{-15}
hzb_15_60mm_450x450x300_sv5	0.05	0.0314 (-0.0179/+0.0249)	0.7878	0.1808	17	4.30×10^{-16}
*hzb_13_60mm_400x400x490_sv1	0.02	0.0395 (-0.0319/+0.0521)	0.6721	0.2284	N/A	N/A
*hzb_12_60mm_400x400x400_sv2	0.10	0.0793 (-0.0479/+0.0608)	0.5784	0.3423	N/A	1.47×10^{-2}
*hzb_12_60mm_400x400x400_sv3	0.10	0.0776 (-0.0474/+0.0606)	0.5923	0.3250	N/A	9.11×10^{-2}
*hzb_14_60mm_400x400x400_sv1	0.20	0.1464 (-0.0546/+0.0590)	0.4646	0.3890	N/A	7.50×10^{-2}
*hzb_14_60mm_400x400x400_sv2	0.20	0.1768 (-0.0598/+0.0629)	0.4377	0.3855	N/A	3.71×10^{-1}
*hzb_15_60mm_400x400x400_sv2	0.05	0.0918 (-0.0334/+0.0368)	0.6684	0.2398	N/A	5.06×10^{-1}
*hzb_15_60mm_400x400x400_sv3	0.05	0.0533 (-0.0266/+0.0325)	0.6630	0.2838	N/A	1.56×10^{-1}
*hzb_15_60mm_400x400x300_sv4	0.05	0.0503 (-0.0259/+0.0332)	0.7017	0.2481	N/A	4.55×10^{-2}
*hzb_15_60mm_400x400x300_sv5	0.05	0.0315 (-0.0179/+0.0249)	0.7869	0.1815	N/A	3.88×10^{-2}
						1.69×10^{-2}

Table 5.1: Summary of harzburgite results. The tag “hzb” stands for harzburgite. Numerical values in the subvolume ID represent the sample name, sample-to-detector distance, subvolume dimensions (measured in voxels), and subvolume identification number, respectively. The superscripts “global” refer to the measured phase volume fractions of each subvolume. d is the geometric mean equivalent diameter. R is the melt partition ratio. (*) Subvolumes were cropped to dimensions that were computationally manageable by FDECC.

to our knowledge, it is the only method available. In principle, the true error can be derived from the point-spread function, would require a ground-truth with a higher resolution 3-D imaging technique.

Since uncertainty in melt fraction is defined by a morphological contraction and dilation of the melt image, a small error in melt fraction requires grains to have a large volume to surface area ratio. Olivine-opx-melt subvolumes have a smaller average grain size than olivine-melt subvolumes and a correspondingly lower volume to surface area ratio. Therefore, uncertainty on melt fraction measurements is higher for olivine-opx-melt subvolumes than for olivine-melt subvolumes.

5.3.2 Permeability

Fluid flow simulations were conducted along the z (vertical) axis of each subvolume. Fig. 5.3 gives the calculated permeabilities of olivine-opx-melt subvolumes as a function of measured melt fraction for each subvolume. Melt fractions were measured for each subvolume by voxel counting. Upper and lower bounds for the uncertainty in the measured melt fraction were computed by applying a 1-voxel dilation and contraction, respectively, to the melt label image (Fusseis et al., 2012). Permeability values were divided by the average grain size squared in order to remove the effect of inter-subvolume grain size variability from permeability. We performed a linear fit to the \log_{10} transform of our data using the total least squares algorithm (York et al., 2004) and plotted it as a solid black line in Fig. 5.3. Before the fit, we applied an ad hoc shift to the measured melt fraction data to compensate for the asymmetric error bars. Partially molten olivine-melt permeability data (Miller et

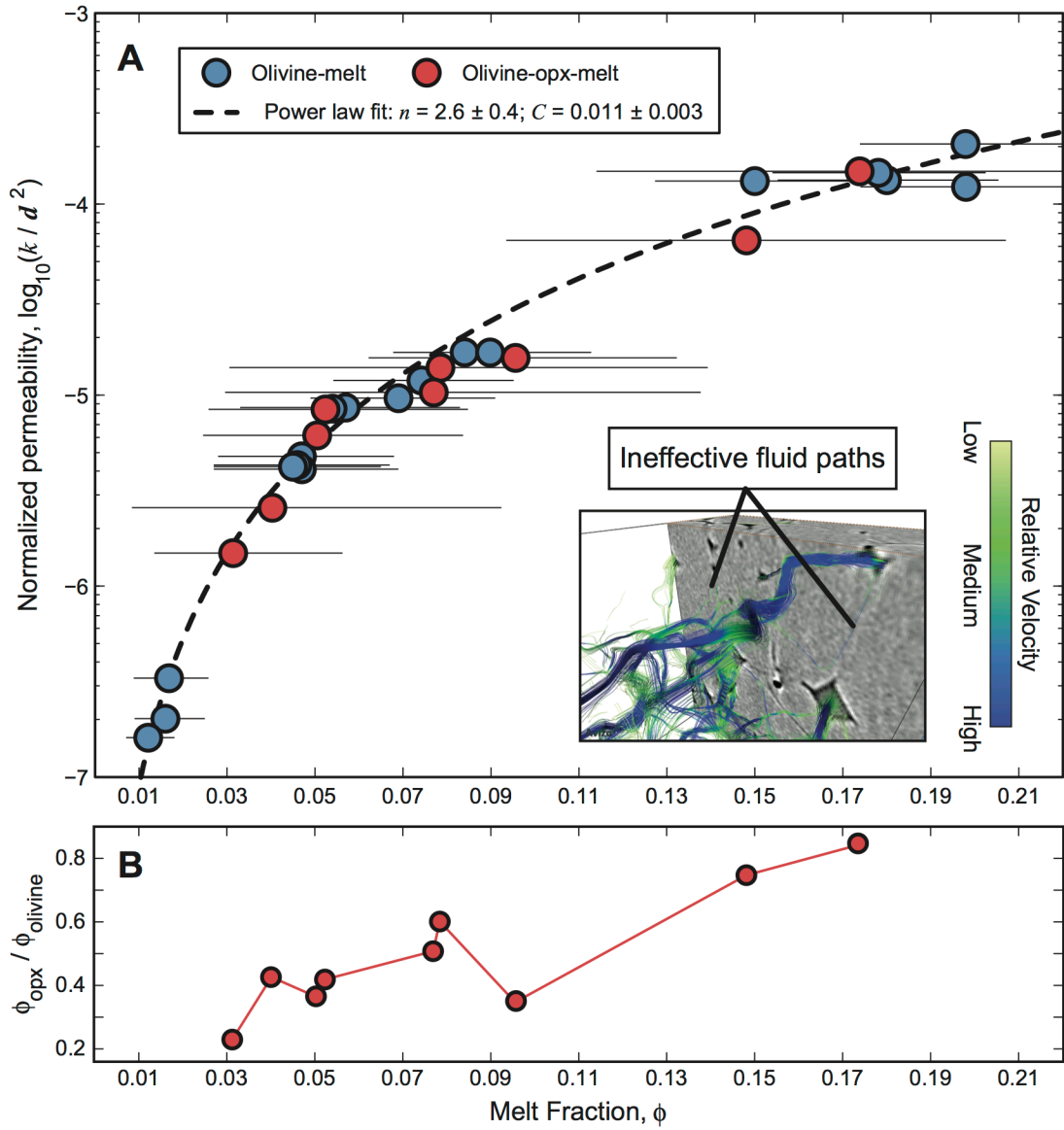


Figure 5.3: Comparison of olivine-melt and olivine-opx-melt subvolume permeabilities. (A) Permeability normalized to the mean equivalent grain diameter plotted as a function of measured melt fraction in each subvolume. Blue data points denote olivine-melt subvolumes and red data points denote olivine-opx-melt subvolumes. Thin, solid black lines represent melt fraction measured for the contracted and dilated melt images. Dotted line represents a fit to the olivine-melt data. Insert is a snapshot of the velocity streamlines, which highlights the efficiency of fluid flow in some conduits (so-called “critical paths”) and inefficiency of fluid flow in others. (B) Proportion of measured olivine to opx measured in olivine-opx-melt subvolumes is plotted as a function of measured melt fraction in order to highlight the discrepancy between the nominal and measured solid phase proportions.

al., 2014) are plotted in Fig. 5.3 for comparison. After normalizing the permeabilities by the mean grain size measured in each subvolume, olivine-opx-melt data plot on the same permeability-melt fraction trend as olivine-melt data, so we conclude the presence of opx does not appear to affect the permeability-melt fraction curve over the melt fractions tested.

5.3.3 Electrical conductivity

Direct current simulations were conducted along the z (vertical) axis of each subvolume. Fig. 5.4 shows the computed bulk electrical conductivities of each subvolume plotted as a function of measured melt fraction. For all direct current simulations, the electrical conductivities of melt and granular phases is assumed to be 7.53 S/m (ten Grotenhuis et al., 2005) and 0.009 S/m (Presnall et al., 1972; Yoshino et al., 2010), respectively. We assumed olivine and opx electrical conductivities are the same. Bulk electrical conductivities of olivine-opx-melt subvolumes were compared to those from partially molten olivine-melt (see Chapter 3 for more details). Archie relations, which are power laws,

$$\sigma_{\text{bulk}} = A\sigma_{\text{melt}}\phi_{\text{measured}}^m \quad (1)$$

were fitted to olivine-melt and olivine-opx-melt subvolume data. In Eqn. (1), σ_{bulk} is the bulk electrical conductivity, σ_{melt} is the melt electrical conductivity. A and m are power law parameters that depend on the spatial distribution of melt. We found that error bounds associated with olivine-opx-melt and the olivine-melt permeabilities overlap but are systematically lower than the dunite bulk electrical conductivities for the same measured melt fraction.

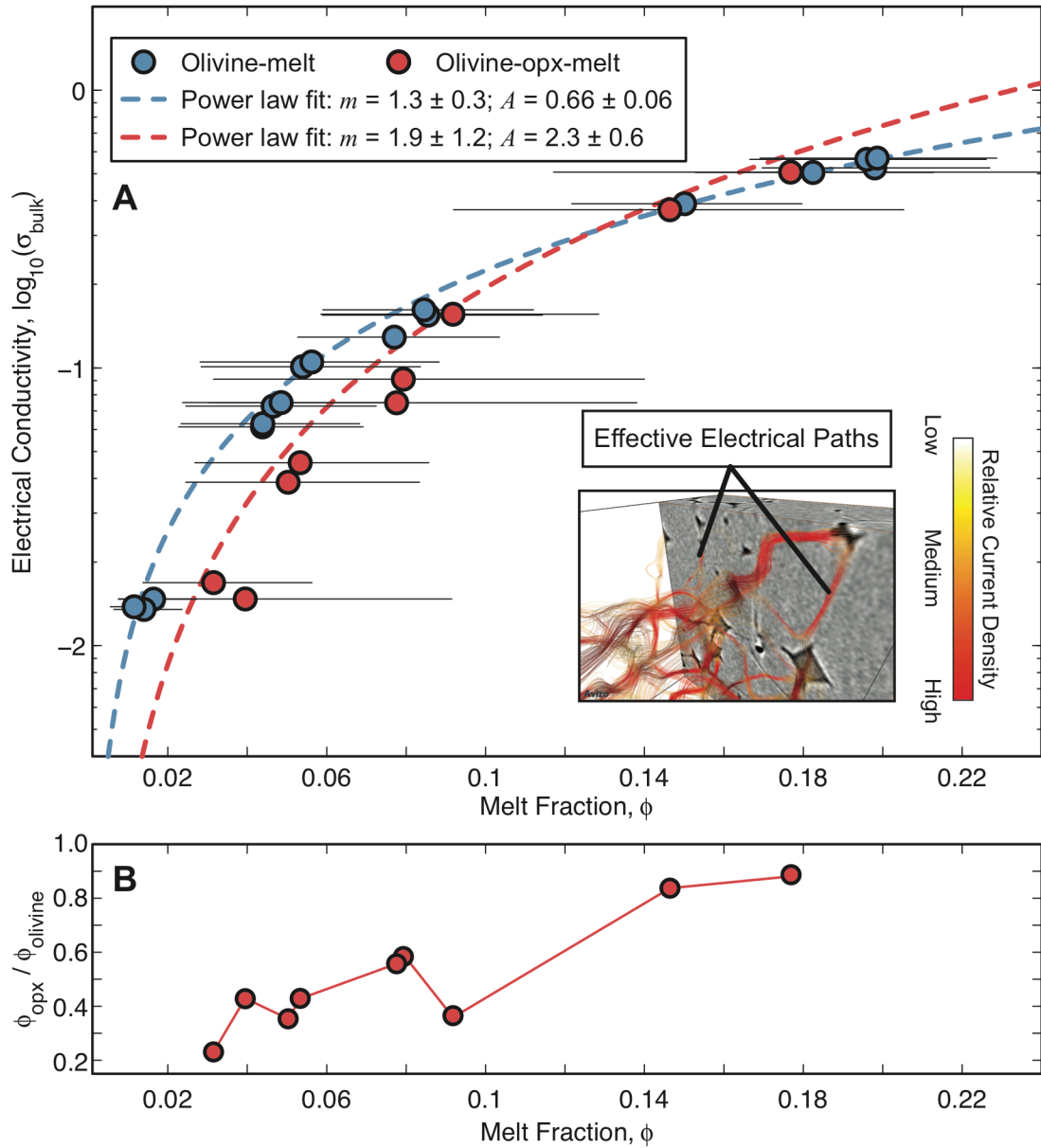


Figure 5.4: Comparison of olivine-melt and olivine-opx-melt subvolume bulk electrical conductivities. (A) Bulk electrical conductivity plotted as a function of measured melt fraction in each subvolume. Blue data points denote olivine-melt subvolumes. Red data points denote olivine-opx-melt subvolumes. Thin, solid, black lines represent estimates of minimum and maximum melt fraction measured for the contracted and dilated melt images, respectively. Dotted lines represent a fits to the olivine-melt (blue) and olivine-opx-melt (red) data. Insert is a snapshot of the electric field streamlines, which highlights the diffusive flow of electricity and the lack of critical paths. (B) Proportion of measured olivine to opx measured in olivine-opx-melt subvolumes is plotted as a function of measured melt fraction in order to highlight the discrepancy between the nominal and measured solid phase proportions.

5.4 Discussion

5.4.1 Influence of opx on permeability

At 3 to 2 olivine to opx ratio, the network permeability models by Zhu and Hirth (2003) suggest there is only a slight reduction in the permeability of olivine-opx-melt subvolumes with respect to olivine-melt subvolumes for all melt fractions. At $\phi = 0.01$, for example, there is only a ~50% reduction in permeability. Though triple junctions along opx grains are less effective conductors of melt flow than those along olivine grains, especially at low melt fraction, the relative insensitivity of permeability to opx (Fig. 5.3A) reflects the tendency for flow to form so-called “critical pathways” (David, 1993; Martys and Garboczi, 1992) in the presence of olivine through which the majority of melt mass is transported. We show evidence for critical pathways in the olivine-melt system in Chapter 2. As opx content increases, the frequency of effective triple junctions decreases. Melt flow reconfigures in response, taking advantage of the remaining viable triple junctions. As a result, permeability decreases only slightly due to the more tortuous pathway (Fig. 5.5A) that it must take to traverse the melt network.

Though we do not see a significant change in permeability from olivine-melt to olivine-opx-melt sample suites, we acknowledge the fact that there is a high degree of variability in the measured olivine to opx volumetric proportions (Fig. 5.3B). Subvolumes that contained smaller melt fraction also have smaller proportions of opx. There are several mechanisms that may account for the correlation between melt and opx proportions, e.g. effects of wetting properties or temperature gradient. Nevertheless, Fig. 5.3B shows us that the threshold opx fraction required to influence

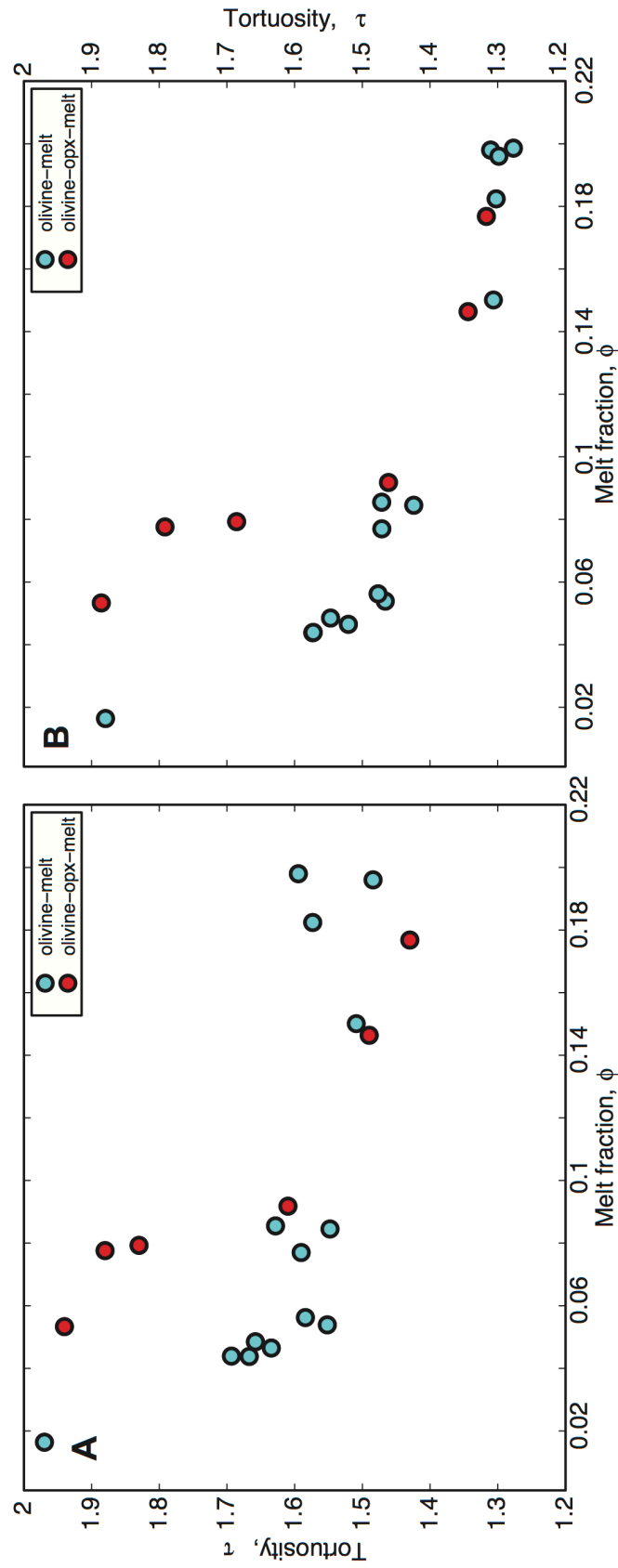


Fig. 5.5: Comparison olivine-melt and olivine-opx-melt tortuosities. (A) Hydraulic tortuosity plotted as a function of measured melt fraction. Values were computed from velocities fields obtained from laminar flow simulations. (B) Electric tortuosity plotted as a function of measured melt fraction. Values were computed from current density fields obtained from direct current simulations.

permeability may not have been attained by the lower melt fraction samples. Network permeability models suggest that at least a 3 to 2 olivine to opx ratio is necessary to reduce permeability. Therefore, in order to conclusively determine the effect of opx on permeability in olivine-opx-melt composite systems, subvolumes having at least 3 to 2 olivine to opx volume ratio and low melt fraction ($\phi < 0.02$) must be examined.

5.4.2 Implications for trace element partitioning in xenoliths

Mineralogical effects on the permeability of mantle rocks may have important implications for interpreting trace element partitioning in peridotite xenoliths. The diffusivity of Li in partially molten mantle rocks is two to three orders of magnitude larger than other trace elements (Richter et al., 2003), making Li a sensitive indicator of melt-rock interactions in the mantle. Studies (e.g. Frey and Green, 1974; Rudnick and Ionov, 2007) observe strong Li disequilibria – both elemental and isotopic – between peridotite xenoliths and the “normal” mantle, which is consistent with an event of mantle metasomatism, i.e. grain-boundary infiltration of a Li-rich melt or fluid (Rudnick and Ionov, 2007). Despite preferential diffusion of Li into clinopyroxene (cpx) over olivine, as evidenced by measured olivine-cpx partitioning coefficients ($^{olivine-cpx}D = 0.2$ to 1.0), refractory harzburgite xenoliths exhibit higher overall enrichment of Li compared to fertile lherzolite xenoliths (Rudnick and Ionov, 2007). One interpretation of this result invokes the wetting properties of peridotite mineral components: if the permeability of olivine-rich (pyroxene-poor) peridotite is higher than olivine-poor peridotite (pyroxene-rich), harzburgite xenoliths will experience higher flux of Li-rich melt than lherzolite xenoliths and thus, higher Li

concentrations. The possibility of using Li as an indicator of permeability is strong motivation for more accurately constraining the permeability of mantle rock at low melt fraction and higher pyroxene content.

5.4.3 Influence of opx on electrical conductivity

Though the permeability-melt fraction relation appears to be unaffected by the presence of opx, the bulk electrical conductivities of olivine-opx-melt geometries are noticeably lower than those of olivine-melt geometries at similar melt fraction (Fig. 5.4A). Contrary to melt percolation, which forms critical pathways due to the high sensitivity of melt flux to the hydraulic radius, electricity conducts more diffusively through the partially molten geometry, increasing the number of viable electrical pathways relative to fluid pathways. Though there are more conduits available for electrical conduction, these “added” pathways are less effective conductors, due to their low hydraulic radius, resulting in reduction of bulk electrical conductivity.

Though there is systematic offset in bulk electrical conductivity between the olivine-melt and olivine-opx-melt suites, we acknowledge there large uncertainties associated with measuring melt fraction from tomographic image data. To better constrain the impact of opx on transport properties beyond what is done in this study, either a better method of characterizing uncertainty associated with measuring phase proportions or a higher-resolution 3-D imaging technique is needed.

5.5 Conclusion

We demonstrated the effect of opx, a low wettability mineral phase that is

common in the upper mantle, on the permeability and electrical conductivity of partially molten mantle rock by conducting numerical simulations of fluid flow and direct current using real rock microstructures. Harzburgite rock samples containing nominal melt fractions of 0.02 to 0.20 and 3 to 2 olivine to opx ratio were synthesized at mantle pressure-temperature conditions. Samples were imaged using X-ray μ -CT and converted to label images to be used as input for numerical computations of permeability and electrical conductivity. We compared transport properties of olivine-opx-melt and olivine-melt aggregates. For the melt fractions examined, we found that harzburgite permeabilities did not deviate from the dunite permeability-melt fraction curve. However, we found that olivine-opx-melt electrical conductivity is lower than olivine electrical conductivity for the same melt fraction, which we interpret by invoking critical pathways for fluid flow. Our data represent the first systematic study that relates macroscopic transport properties of partially molten mantle rocks containing more than one mineral phase to rock microstructure.

Chapter 6: Summary and future work

6.1 Summary of results and conclusions

This dissertation work represents a significant achievement in the linking of macroscopic material properties of partially molten mantle rock to microstructural characteristics. Previous attempts to characterize permeability of partially molten mantle rocks, for the most part, rely on 2-D images of partially molten rocks to infer permeability, which is intrinsic to the 3-D melt microstructure and are therefore inadequate. However, recent advances in X-ray imaging technology allow us to capture, in high-resolution, the 3-D microstructure of partially molten rocks. These images constitute digital rock samples on which any number of non-destructive virtual rock physics experiments can be conducted. These so-called digital rock physics (DRP) simulations are fast, accurate, and repeatable (Andrä et al., 2013) and enable the user to straightforwardly conduct rock physics experiments without having to devise elaborate experimental systems.

Over the course of this project, we developed a number of tools for automatically quantifying the microstructure and transport properties of our digital samples. For example, we were able to quantify, by skeletonizing our melt geometry, the interconnectivity of melt network as a function of melt fraction and sintering duration. Though it is not discussed in this document, skeletonized melt networks can also be used in network models to compute permeability and electrical conductivity. An automatic grain separation algorithm allowed us to characterize the grain size distributions of our samples without having to infer a 3-D grain-shape; though there

were some approximations made about the location of grain-grain boundaries. Furthermore, we are able to use the 3-D geometry as direct input to numerical models to compute permeability and electrical conductivity as a function of melt fraction.

Using a combination of experimental petrology, conventional rock physics, advanced imaging, and numerical modeling, we were able to formulate meaningful empirical permeability-melt fraction and electrical conductivity-melt fraction relations. Our permeability-melt fraction relation confirms the rate at which melt separates from residue, a critical parameter in multiphase flow models of melt transport at mid-ocean ridges. A simple 1-D model, suggests that, with the new permeability-melt fraction relation, estimates of melt fraction in the upper mantle inferred from U-series geochemistry are more or less consistent with those inferred from geophysical datasets. The electrical conductivity-melt fraction relation we presented will be used in future studies to guide better interpretation of electromagnetic data. A side-by-side comparison of fluid flow and direct current on the same melt geometries determined that fluid and electricity have different sensitivities to the pathways available to flow. We argued, based on first principles, that, aside from an empirical similarity, there is no evidence that a rigorous link between permeability and electrical conductivity exists.

Our DRP approach allowed us to test the influence of opx, a low wettability mineral that is common in the upper mantle, on transport properties of partially molten mantle rock. Before this study, the only evidence opx affected transport properties came from synthetic datasets (Zhu and Hirth, 2003) and 2-D microscopy analysis of synthetic samples composed of olivine and basalt. Using the tools

described in this thesis, we confirmed that spatial variations in the surface energy distribution, related to the presence of opx, caused lithologic melt partitioning. Lithologic melt partitioning did not appear to alter the permeability of our samples over the melt fractions tested. However, if combined with another mechanism that creates a parallel mineralogical structure, such as the reaction infiltration instability, lithologic melt partitioning may increase the efficiency of melt transport in the upper mantle.

6.2 Future research directions

We have just scratched the surface in what we can do with DRP. Potential future directions include experiments with deformed samples, eigenfrequency analysis of electrical conductivity, and evaluation of seismic properties of partially molten rocks.

The upper mantle is a dynamic system (Turcotte and Schubert, 2014). Experiments and models suggest that there is a coupling between shear deformation and porous flow that give rise to high-melt fraction bands (Daines and Kohlstedt, 1997; Holtzman and Kohlstedt, 2007; Holtzman et al., 2003; King et al., 2011a; King et al., 2011b; Qi et al., 2014; Zimmerman et al., 1999). These bands may play an important role in melt transport and melt focusing at mid-ocean ridges. In order for permeabilities derived from synthetic partially molten rocks to be directly applicable to the upper mantle, sheared samples must be considered.

Experimental determination of electrical conductivity through impedance spectroscopy is technically challenging because there may be different conduction

mechanism, e.g. conduction through the sample, conduction through the pressure vessel, conduction through some surface layer, that operate at the same frequency spectrum (Nover, 2005; Yoshino, 2010). A numerical impedance spectroscopy analysis of our images can be used to deconvolute those various processes and help to interpret experimental results. Specifically, a comparison between numerical and experimental impedance spectroscopy can be used to test the hypothesis that there is surface conduction through an electrical double layer at the grain-melt interface that contributes to the bulk conductivity of the aggregate. However, the fairest comparison between experiments and digital rock physics simulations would involve imaging the samples that were used in actual impedance spectroscopy experiments.

It would be of tremendous value to the seismology community studying seismic wave propagation at mid-ocean ridges or subduction zones to use DRP techniques to constrain the bulk modulus of partially molten rock as a function of melt fraction. As a first approach, static loading models conducted on the 3-D melt geometries to reduce the error of 2-D models (e.g. Hammond and Humphreys, 2000). Though software needs to be developed to handle many degrees of freedom associated with static loading models on large subvolumes. Eventually, wave-propagation codes, similar to Saenger and Bohlen (2004), can be used to determine frequency dependence of partially molten mantle rock; though significant advances need to be made in the modeling of grain boundary slide as a mechanism for energy dissipation.

Appendix A: Supplementary information for microstructure and permeability quantification

A.1 Removing noise using anisotropic diffusion filtering

We used an *edge-preserving smoothing* filter to remove noise from our tomography data. This particular algorithm is an implementation of anisotropic diffusion (Weickert et al., 1998) and is provided as part of the Avizo image filter library. Anisotropic diffusion is a class of smoothing filters that reduces noise by numerically solving the three-dimensional diffusion equation,

$$\frac{\partial I(\mathbf{x}, t)}{\partial t} = \nabla \cdot \left[\mathbf{D} \left(\left| \nabla I(\mathbf{x}, t) \right|^2 \right) \nabla I(\mathbf{x}, t) \right] \quad (\text{A1})$$

where I is the position (\mathbf{x}) and time (t) dependent scalar field representing the grayscale pixel intensity and \mathbf{D} is the diffusivity tensor, which is a function of the local intensity gradient squared. Stepping in time, each image is given as a convolution of the previous image and a diffusivity kernel.

For a constant diffusivity, Eqn. (A1) is linear, and the problem is equivalent to a Gaussian blur. Linear diffusion filters are effective at removing random noise from the tomography data; however, diffusion occurs without any a priori information about the image, often costing edge resolution. In our samples, where the phase contrast is low, edges are often the only distinguishing feature in the data. Therefore, it is vital that we preserve fine details in the tomography images, such as phase boundaries.

We employ an anisotropic diffusion filter (Fig. A.1). Anisotropic diffusion uses information about the local grayscale intensity gradient, which is known a priori,

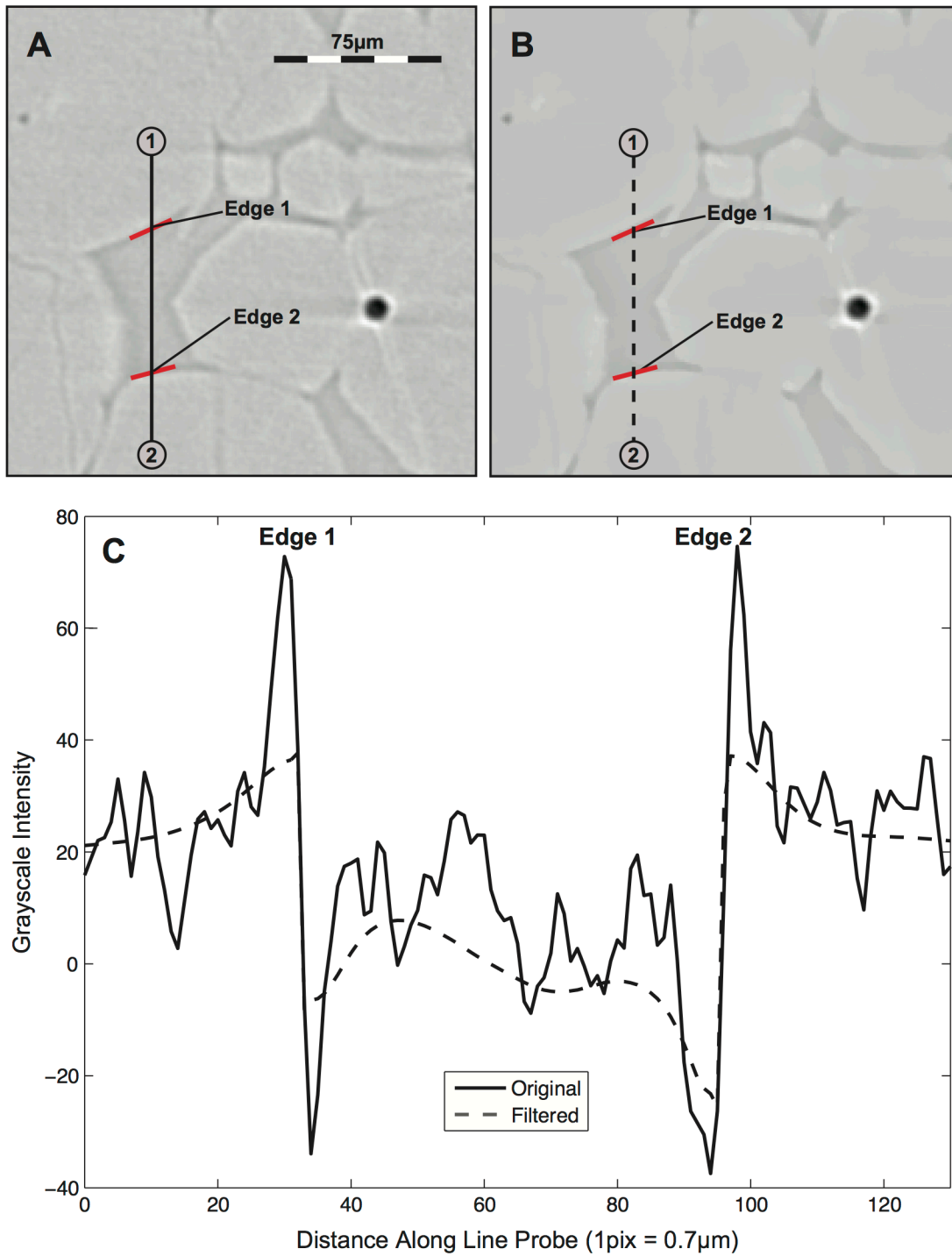


Figure A1: Removing noise from $200 \times 200 \times 200 \mu\text{m}^3$ scoba-5 (melt fraction = 0.20) subvolume by implementing an anisotropic diffusion filter. A) Original (noisy) image. B) Filtered (denoised) image. Olivine grains appear to be lighter gray and melt phase is darker gray. Grain-melt interfaces (i.e., edges) are partially highlighted in red. C) Comparison of grayscale intensities along profile (1)-(2) from 0 to 125 in A) (solid curve) and B) (dashed curve). Parameters for anisotropic diffusion were 25s for the total diffusion time, 1s for the time interval, and 65 for the diffusion threshold.

to prevent diffusion across edges. This is accomplished by defining a threshold value c in the formulation of an anisotropic diffusivity kernel that limits diffusion between pixels whose intensities differ by I_C . Correctly calibrating I_C ensures that diffusion does not occur over edges, leaving well-defined phase boundaries.

Other parameters of the anisotropic diffusion filter include the total diffusion time and time step. As a general rule, the shorter the time step, the more accurately the diffusion equation is solved. However, shortening the time step necessitates a longer computation and may cause issues of solver stability. For our purposes, a total diffusion time of 25s and time step of 1s yields good results within an acceptable timeframe. Regarding the threshold, values of I_C typically range between 35 and 75 when range of grayscale values over the whole image is -500 to 500.

Fig. A.1 illustrates one application of the anisotropic diffusion filter to a $200 \times 200 \times 200 \mu\text{m}^3$ subvolume. The application of anisotropic diffusion results in a smoother, less noisy image than the original that is largely free of artifacts, such as streaks. The resulting image is also better conditioned for global thresholding than the original image.

A.2 Segmenting using watershed transformation

The Avizo[®] watershed transformation algorithm was implemented for segmenting data with small phase contrast. We start with a grayscale image processed by the anisotropic diffusion filter described above (Fig. A.2A) and compute the gradient magnitude of pixel intensity. Due to the edge-enhanced imaging technique, the highest gradients in our olivine-basalt samples occur at grain edges (Fig. A.2B). A

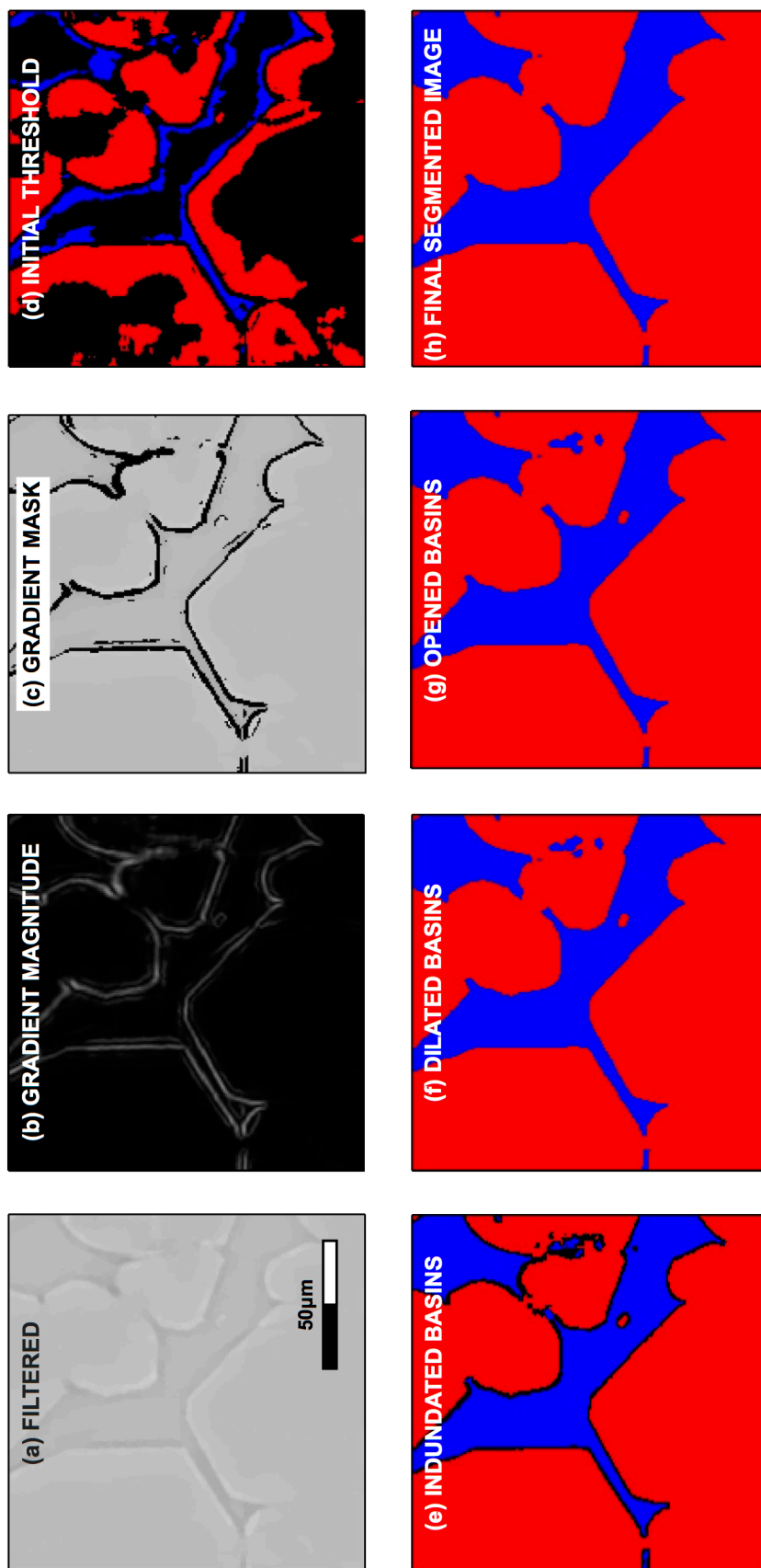


Figure A2: Schematic of the steps involved in implementing the watershed segmentation algorithm. (a) Filtered grayscale image. Olivine grains appear lighter gray and melt phase is darker gray. (b) Grayscale gradient magnitude of filtered image: white represents high gradient region and black represents low gradient region; (c) Thresholded higher-end values (phase interface) of the gradient magnitude masking the filtered grayscale image; (d) Initial inundation based on a global threshold: olivine (red), melt (blue), and yet to be determined phase (black); (e) Fully inundated watershed basins. (f) Basins dilated to remove phase fine details of the boundary region; (g) Opening filter applied to the dilated image to smooth small details at phase boundaries; (h) Manually refined binary image. Phase boundaries are defined at the grayscale inflection of (a).

global threshold was then applied to the gradient magnitude image to record the positions of pixels located within the phase transition regions. This is called the gradient mask (Fig. A.2C). Next, an initial inundation is marked using a global threshold where phases are unambiguously defined (Fig. A.2D). The watershed transform is then applied. Flooding begins from the initial inundation and continues until meeting the gradient mask (Fig. A.2E). The gradient mask acts as an impermeable barrier through which different flooding regions cannot spill into one another. The watershed transformation is analogous to flooding drainage basins in natural watershed systems, hence the name of the algorithm. The labeled basins were then dilated to fill the defined gradient mask (Fig. A.2F). Once segmented, a 3-D *opening* filter was applied to the binary data, which removed small details at boundaries and opened passages separated by only two pixels (Fig. A.2G). Some small manual adjustments (e.g. hole filling) were often needed to produce accurate segmentations. The final result after the watershed transformation is a high-quality, binary image where phase boundaries are defined exactly at grayscale inflections (Fig. A.2H).

A.3 Determining the size of the representative volume element

Because of heterogeneity in melt distribution, permeability may depend on the size of the subvolume. In order to determine the minimum subvolume size that represents a statistically significant portion of the sample, we computed the permeability of several subvolumes cropped from scoba-12 ($\phi_n = 0.05$) ranging in size from $140 \times 140 \times 140 \mu\text{m}^3$ to $350 \times 350 \times 350 \mu\text{m}^3$. The APES module was used for

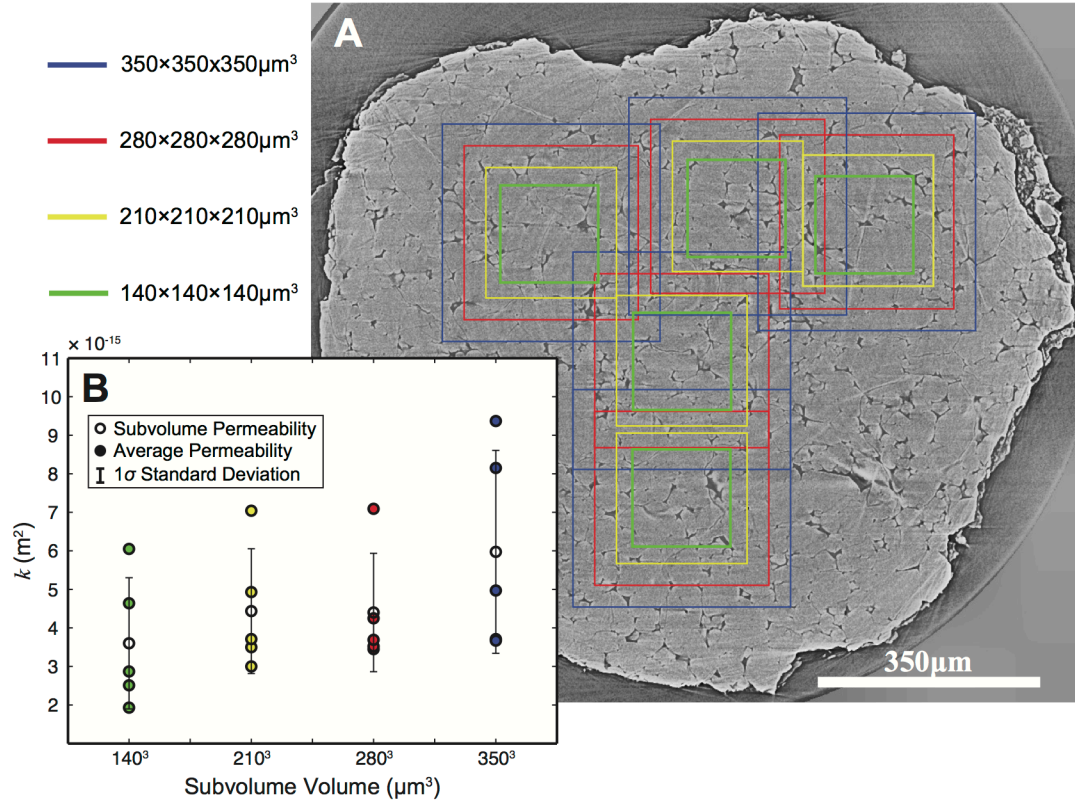


Figure A3: (A) Cross-section of scoba-12 (melt fraction is 0.05) as seen in the reconstructed tomography data taken perpendicular to the long axis of the cylindrical sample. Permeability was calculated for several subvolumes of various sizes (colored circles) and plotted against subvolume volume in order to determine the minimum subvolume size that is adequately representative of the whole sample. Also plotted are the average permeabilities pertaining to each size category of subvolumes (black, unfilled circles). Bars represent the 1σ standard deviation and are present to visualize the spread in the data. Boxes in the grayscale image represent cubic sample regions where permeability computations were performed. The color of the box is indicative of its size, e.g. blue--350×350×350 μm³, red--280×280×280 μm³, yellow--210×210×210 μm³, and green--140×140×140 μm³.

permeability computations (Fig. A.3A). We plot their permeability as a function of subvolume volume (Fig. A.3). Permeability values of different subvolumes of similar size are consistent within a factor of 4. The average permeability of each subvolume group and the standard deviation (1σ) are reported in Fig. A.3. Compared to the spread in permeability values, which are a result of region-to-region variation, the size of the simulation domain has little effect on simulation results (Table A.1). Based on these results, we consider the permeability calculations performed in this study to be representative of the bulk sample from which they were cropped. To guarantee that our results are representative of the sample, we only report in the main text permeability calculated on the largest possible subvolume size ($350 \times 350 \times 350 \mu\text{m}^3$).

A.4 Cleaning the skeletonized melt network

Much like segmentation, there are artifacts that arise from the thinning algorithm during the skeletonization of the melt network. Some of these artifacts include clusters of nodes and short channels where there should be a single junction. These artifacts typically occur at large melt pools or at wetted grain boundaries. In histograms of connectivity, these artifacts manifest as anomalously high numbers of the coordination number 3 nodes, where coordination number refers to the number of edge connections possessed by a node.

A Matlab[®] script, called *ScobaCleaner.m*, was written for automatically removing artifacts in the skeleton network. Four types of artifacts exist. They are denoted loops, sublinks, twins, and short-links (Table A2). Inevitably, there will be

some short-links that should not be merged, sometimes resulting in the formation of a artificial high-connectivity junction (Zhu et al., 2011).

A.5 Time series experiment

In order to evaluate the time necessary to achieve textural equilibrium, we created a time series of charges. The nominal melt fraction is 0.05 for those charges, and the sintering durations are 42 hours (scoba-13), 84 hours (scoba-14), 168 hours (scoba-12), and 336 hours (scoba-15), respectively. A low nominal melt fraction was chosen for the time series experiments because low melt fraction samples take longer to equilibrate than higher melt fraction ones (Cmíral et al., 1998), which gives us a maximum estimate for the time required for our samples to reach textural equilibrium. A melt fraction 0.05 is best choice given the current resolution limitation of μ -CT.

Equivalent diameter distributions (EDD) were computed for $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes from scoba-13-500-2, scoba-14-500-1, scoba-12-500-1, and scoba-15-500-1 (Fig. A.4). The EDD's of scoba-13-500-2 and scoba-14-500-1 (Figs. A.4A and A.4B), which are shorter duration experiments, differ substantially from scoba-12-500-1 and scoba-15-500-1. The longer duration charges scoba-12 and scoba-15 have nearly identical EDDs (Figs A.4C and A.4D), suggesting that grain size evolution has reached an essentially steady state. The similarity between the two longer duration experiments suggests that textural equilibrium (Wark and Watson, 1998) is reached approximately some time between 84 and 168 hours for olivine-basalt aggregates with a nominal melt fraction of 0.05. The similarity between the

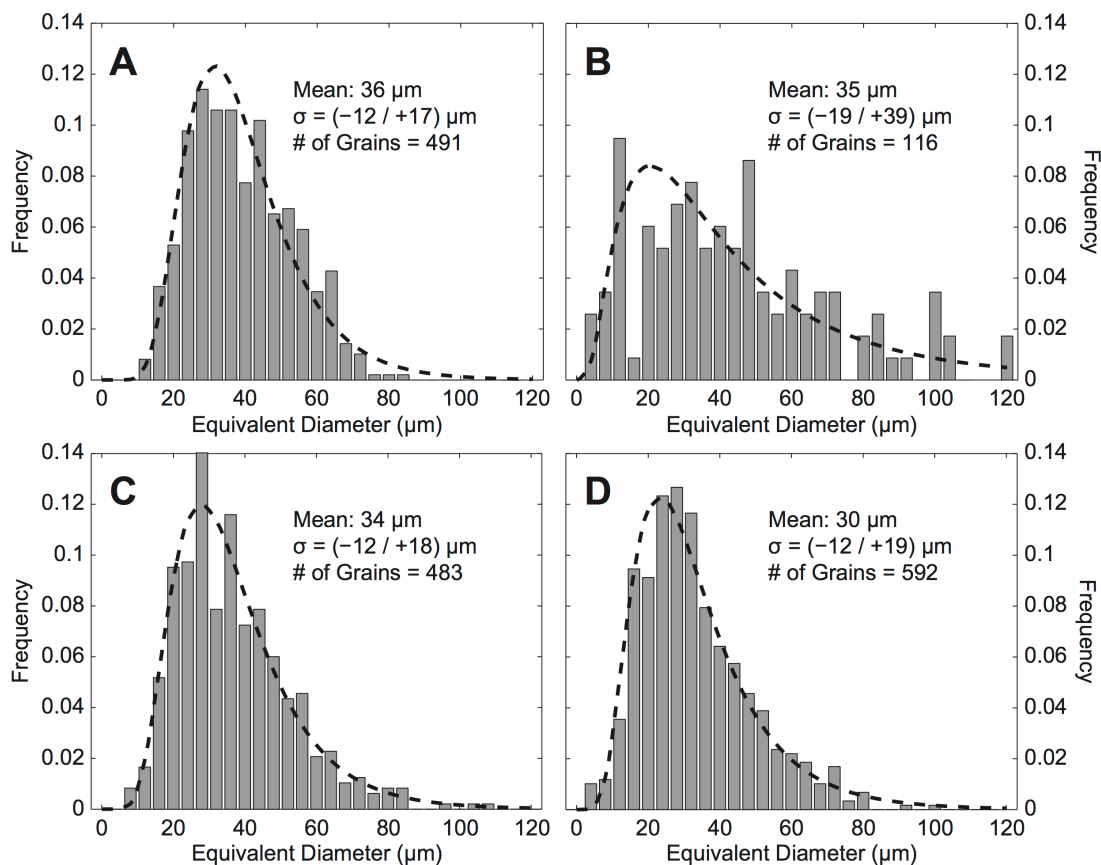


Figure A4: Equivalent diameter distributions from $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes of A) scoba-13-500-2 (42 hours), B) scoba-14-500-1 (84 hours), C) scoba-12a-500-1 (168 hours), and D) scoba-15-500-1 (336 hours). The geometric mean, error bounds from the geometric standard deviation, and number of grains contained within the subvolume are reported. Dashed lines represent the best-fit lognormal distributions to the grain size data.

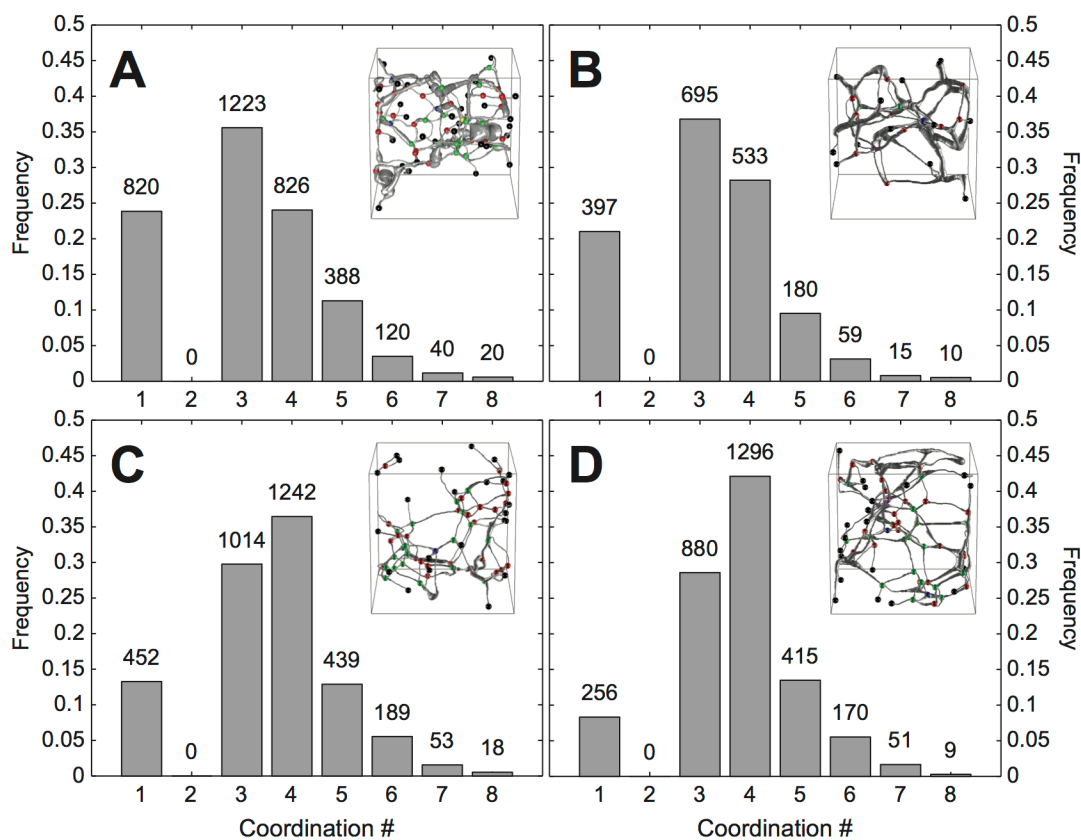


Figure A5: Coordination number distribution of time series experiments. Frequency of coordination numbers from $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes of A) scoba-13-500-2 (42 hours), B) scoba-14-500-1 (84 hours), C) scoba-12a-500-1 (168 hours), and D) scoba-15-500-1 (336 hours), respectively. Total counts of each coordination number's appearance within the subvolume are reported. The skeletonized melt networks of a representative subvolume of dimensions $105 \times 105 \times 105 \mu\text{m}^3$ is included as insert for each sample. The nodes in the skeleton are color-coded according to their coordination number, e.g. 1-black, 3-red, 4-green, 5-blue, 6-magenta, and > 6 -yellow. The radius of melt tubules in the skeletonized network are in proportion to the thickness of melt conduits.

Table A.1: Summary of simulation results.

Subvolume ID	ϕ_m^a	x -Position ^b	y -Position ^b	z -Position ^b	Volume [pixels]	k_c [μm^3]	ϕ_m^d	d_c [μm]	Segmentation Method ^f
scoba-15-500-1	0.05	718-1217	1061-1560	521-1020	(500) ³	7.74×10^{-3}	0.057 (-0.024 / +0.026)	30 (-11 / +19)	global threshold
scoba-12a-500-1	0.05	499-998	899-1398	500-999	(500) ³	4.73×10^{-3}	0.047 (-0.020 / +0.022)	34 (-12 / +18)	top-hat
scoba-12a-400-1	0.05	549-948	949-1348	550-949	(400) ³	4.70×10^{-3}	0.049 (-0.020 / +0.023)	N/A	top-hat
scoba-12a-300-1	0.05	599-898	999-1298	600-899	(300) ³	4.52×10^{-3}	0.049 (-0.020 / +0.023)	N/A	top-hat
scoba-12a-200-6	0.05	649-848	1049-1248	650-849	(200) ³	3.44×10^{-3}	0.047 (-0.019 / +0.021)	N/A	top-hat
scoba-12a-500-2	0.05	1238-1737	919-1418	500-999	(500) ³	8.15×10^{-3}	0.054 (-0.022 / +0.025)	31 (-11.2 / +18)	global threshold
scoba-12a-400-2	0.05	1288-1687	969-1368	550-949	(400) ³	7.09×10^{-3}	0.053 (-0.022 / +0.025)	N/A	global threshold
scoba-12a-300-2	0.05	1338-1637	1019-1318	600-899	(300) ³	7.04×10^{-3}	0.055 (-0.022 / +0.025)	N/A	global threshold
scoba-12a-200-5	0.05	1388-1587	1069-1268	650-849	(200) ³	6.05×10^{-3}	0.048 (-0.019 / +0.021)	N/A	global threshold
scoba-12a-500-3	0.05	828-1327	260-759	500-999	(500) ³	4.40×10^{-3}	0.046 (-0.019 / +0.021)	32 (-12 / +19)	top-hat
scoba-12a-400-3	0.05	878-1277	310-709	550-949	(400) ³	4.02×10^{-3}	0.044 (-0.018 / +0.021)	N/A	top-hat
scoba-12a-300-3	0.05	928-1226	360-659	600-899	(300) ³	3.17×10^{-3}	0.044 (-0.019 / +0.021)	N/A	top-hat
scoba-12a-200-7	0.05	977-1176	410-609	650-849	(200) ³	2.09×10^{-3}	0.045 (-0.019 / +0.021)	N/A	top-hat
scoba-12a-500-4	0.05	828-1327	605-1104	500-999	(500) ³	4.59×10^{-3}	0.045 (-0.018 / +0.020)	33 (-11 / +18)	global threshold
scoba-12a-400-4	0.05	878-1277	655-1054	550-949	(400) ³	3.69×10^{-3}	0.044 (-0.018 / +0.020)	N/A	global threshold
scoba-12a-300-4	0.05	928-1227	705-1004	600-899	(300) ³	3.50×10^{-3}	0.045 (-0.019 / +0.022)	N/A	global threshold
scoba-12a-200-8	0.05	97-1177	755-954	650-649	(200) ³	2.87×10^{-3}	0.045 (-0.018 / +0.021)	N/A	global threshold
scoba-12a-500-5	0.05	938-1437	960-1459	500-999	(500) ³	4.89×10^{-3}	0.047 (-0.019 / +0.021)	32 (-11 / +17)	top-hat
scoba-12a-400-5	0.05	988-1387	1010-1409	550-949	(400) ³	4.25×10^{-3}	0.047 (-0.019 / +0.022)	N/A	top-hat
scoba-12a-300-5	0.05	1038-1337	1060-1359	600-899	(300) ³	4.93×10^{-3}	0.048 (-0.020 / +0.022)	N/A	top-hat
scoba-12a-200-9	0.05	1088-1287	1110-1309	650-849	(200) ³	4.64×10^{-3}	0.0442 (-0.0214 / +0.0207)	N/A	top-hat
scoba-5-500-1	0.20	809-1308	879-1378	760-1259	(500) ³	2.24×10^{-1}	0.1800 (-0.0248 / +0.0254)	41 (-15 / +24)	watershed + top-hat
scoba-5-500-2	0.20	749-1248	1112-1611	1036-1535	(500) ³	1.71×10^{-1}	0.1500 (-0.0227 / +0.0233)	36 (-14 / +24)	watershed + top-hat
scoba-5-500-3	0.20	749-1248	732-1231	1036-1535	(500) ³	2.82×10^{-1}	0.1979 (-0.024 / +0.0245)	37 (-15 / +27)	watershed + top-hat
scoba-5-500-4	0.20	928-1427	615-1114	370-869	(500) ³	2.45×10^{-1}	0.1781 (-0.0241 / +0.0244)	41 (-21 / +42)	watershed + global thresh.
scoba-5-500-5	0.20	838-1337	702-1201	510-1009	(500) ³	2.60×10^{-1}	0.198 (-0.024 / +0.025)	46 (-16 / +25)	watershed + top-hat
scoba-6-500-1	0.10	359-558	582-1081	1292-1791	(500) ³	1.18×10^{-2}	0.069 (-0.020 / +0.022)	35 (-13 / +20)	watershed + top-hat
scoba-6-500-2	0.10	479-978	471-970	640-1139	(500) ³	2.29×10^{-2}	0.084 (-0.021 / +0.020)	37 (-14 / +22)	watershed + top-hat
scoba-6-500-3	0.10	838-1377	160-659	900-1399	(500) ³	2.54×10^{-2}	0.0898 (-0.022 / +0.023)	39 (-15 / +26)	watershed + top-hat
scoba-6-500-4	0.10	698-1197	88-587	460-959	(500) ³	1.72×10^{-2}	0.0742 (-0.020 / +0.021)	38 (-13 / +21)	watershed + top-hat
scoba-9-500-2	0.02	579-1078	552-1051	470-969	(500) ³	5.27×10^{-4}	0.0168 (-0.008 / +0.009)	40 (-19 / +37)	top-hat
scoba-9-500-3	0.02	589-1088	737-1236	1290-1789	(500) ³	3.56×10^{-4}	0.0160 (-0.007 / +0.009)	42 (-20 / +38)	top-hat
scoba-9-500-4	0.02	579-1078	1134-1633	460-959	(500) ³	3.55×10^{-4}	0.0121 (-0.005 / +0.006)	47 (-23 / +43)	top-hat
scoba-13-500-2	0.05	309-808	943-1442	50-549	(500) ³	N/A ^g	N/A	36 (-12 / +17)	global threshold
scoba-14-500-1	0.05	480-979	1325-1824	500-999	(500) ³	N/A	N/A	35 (-19 / +39)	global threshold

^aNominal melt fraction. ^bCoordinates of subvolume within sample. ^cMeasured permeability of subvolume. ^dMeasured melt fraction of subvolume.

^eMeasured geometry mean equivalent diameter. ^fMethod used for segmenting grayscale data. All subvolumes required some degree of user-controlled refinement in addition to the automatic segmentation algorithm used. ^gN/A is listed if no computation performed for that subvolume.

mean equivalent diameters suggests that grain growth was probably very slow after 42 hours.

Coordination number distributions (CND) were also computed for $350 \times 350 \times 350 \mu\text{m}^3$ subvolumes from scoba-13-500-2, scoba-14-500-1, scoba-12-500-1 and scoba-15-500-1 (Fig. A.5). Comparison of the CNDs of these samples reveals that the number of dead-end nodes with coordination number of 1 decrease with increasing sintering time. Nodes with coordination number of 3 are mostly associated with regions where melt pooling or grain boundary wetting is occurring. We observe an inversion between the frequency coordination number 3 and 4 nodes, indicating a migration of the melt from grain boundaries to tubules. In subvolume cubes scoba-12-500-1 and scoba-15-500-1, nodes with coordination number of 4 are the most abundant, which is consistent with the idealized model of an isotropic system at textural equilibrium (von Bargen and Waff, 1986). Though there is a small increase in the relative abundance of coordination number 4 nodes from 168 hours to 336 hours, the melt network appears to have reached an approximately steady state by 168 hours.

A.6 Correcting for skeletonization artifacts

Main artifacts during skeletonization and corrections. The skeleton network is a simplified representation of the complex melt microstructure. Included in the table are visualizations of the skeleton artifacts. Edges and nodes in question are highlighted in yellow. All other edges and nodes are colored gray and red, respectively. Actions taken by *ScobaCleaner* for simplifying the skeletonized melt network and the effect on the coordination number distribution are summarized.








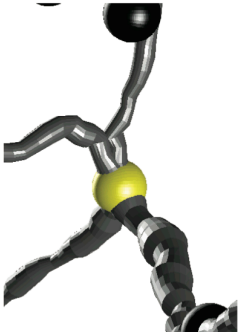
Artifact	Illustration	Action	Illustration	Result of Action
Loop		If loop is very short (~ 1 pixel long), loop is removed. Otherwise, loop is split into two segments of approximately equal length, and collapses them into a single segment that has the same volume of the original loop.		If loop is removed, reduces coordination # of connecting node by $n - 1$. Otherwise, a coordination #1 node is created and added to the network.
Sublink		Shared ends of two segments are connected to form a single segment that has the same volume as the original two segments.		Removes nodes whose coordination # equals two from the connectivity histogram.
Twin		Collapses segments that are shared between two nodes into a single segment that has the same volume as the original two segments.		If there are n twins between two nodes, the coordination # of each node is reduced by $n-1$.
Short-link		Removes segments shorter than $7\mu\text{m}$ by merging the junctions associated with the end nodes.		If n_1 and n_2 are the coordination numbers associated with the nodes of the short-link, the coordination # of the final node n_f is $n_f = n_1 + n_2 - 2$.

Table S2: Actions taken by ScobaCleaner for simplifying the skeletonized melt network. Main artifacts during skeletonization and corrections. The skeleton network is a simplified representation of the complex melt microstructure. Included in the table are visualizations of the skeleton artifacts. Edges and nodes in question are highlighted in yellow. All other edges and nodes are colored gray and red, respectively. Actions taken by ScobaCleaner for simplifying the skeletonized melt network and the effect on the coordination number distribution are summarized.

```

% ----- %
function [Edge, Node] = SkeletonWrapper(directory, fname, dim, lt, varargin)

%
% -----
% This is the wrapper script for pruning skeletonized tomography data.
% Several artifacts, which often arise from skeletonization but are not
% real features that appear in the binary image data, are removed with this
% algorithm. They are:
%
% 1) Loops -- Edges that form a loop
% 2) Sublinks -- two edges with one node connecting them where that
%    node does not have any other connections.
% 3) Short Edges -- edges whose length is less than the input lt
% 4) Islands -- Nodes that do not have any connecting edges or single
%    edges that are not connected to the rest of the network.
%    that are not connected to the rest
% 5) Twins -- twin edges that share the same node endings
%
% More info about how these artifacts are removed from the skeleton network
% is given in the online supplement of Miller et al. (2014) in Earth and
% Planetary Science Letters and Zhu et al. (2011) in Science.
%
% Miller, K.J., Zhu, W., Montési, L.G.J., Gaetani, G. A., 2014.
% Experimental quantification of permeability of partially molten mantle
% rock. Earth Planet. Sci. Lett. 388, 273-282.
%
% Zhu, W., Gaetani, G.A., Fosseis, F., Montési, L.G.J., De Carlo, F., 2011.
% Microtomography of partially molten rocks: three-dimensional melt
% distribution in mantle peridotite. Science 332, 88-91.
%
% Inputs:
%
% 'directory' --> (string) directory where skeleton text file is
%                located
% 'fname'      --> (string) name of skeleton text file
% 'dim'        --> (number of any precision) vector specifying the x,
%                y, and z dimensions of the skeleton
% 'lt'         --> (number of any precision) desired maximum length of
%                edges. Edges whose length is lower than lt are
%                preserved, while those larger than lt are pruned
% 'varargin'   --> (cell) variable input parameter that contains the
%                following inputs.
% 'Print'      --> (string) Prints initial and pruned results to pdf
%                file specified by the string immediately following
%                'Plot'. Warning: Case-sensitive!
% 'Save'       --> (string) Saves the pruned results 'Node' and 'Edge'
%                to .mat files specified by string immediately
%                following 'Save'. Warning: Case-sensitive. '-Edge'
%                and '-Node' are appended to the ends of file name.
%
% Outputs:
%
% 'Edge'       --> (structure) Structure that contains position,
%                connectivity, and thickness information associated
%                with edges.
% 'Node'       --> (structure) Structure that contains position and
%                connectivity information about nodes, where edges
%                are connected.
%
% Example:
%
% Run cleaning algorithm for skeleton 'sample_1_skeleton.txt'
% Removes edges longer than 10 length units. Saves and prints those
% results. Skeleton data is stored as text file, which is outputted by
% Avizo.
%
% [Edge, Node] = skeletonwrapper( ...

```



```

        nc(1), ...
        nc(2), ...
        nc(3), ...
        nc(4), ...
        nc(5), ...
        nc(6), ...
        nc(7), ...
        nc(8), ...
        nc(9), ...
        nc(10), ...
        length(Node) ...
    );

% removing island edges from model, since these do not conduct flow
[Edge, Node] = RemoveIslands(Edge, Node);

% Printing connectivity after removal of island edges and nodes
[nc, ~] = hist([Node.connectivity], 1:10);
fprintf('%s|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i\n', ' RI ', ...
        nc(1), ...
        nc(2), ...
        nc(3), ...
        nc(4), ...
        nc(5), ...
        nc(6), ...
        nc(7), ...
        nc(8), ...
        nc(9), ...
        nc(10), ...
        length(Node));

% main loop that removes loops, sublinks, and twin edges
[Edge, Node, ~] = MainLoop(Edge, Node);

[nc, ~] = hist([Node.connectivity], 1:10);
fprintf('%s|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i\n', ' ML ', ...
        nc(1), ...
        nc(2), ...
        nc(3), ...
        nc(4), ...
        nc(5), ...
        nc(6), ...
        nc(7), ...
        nc(8), ...
        nc(9), ...
        nc(10), ...
        length(Node) ...
    );

fprintf('-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----\n');

vt_vec = [];

% removes short edges by absorbing their volume into neighboring edges
[Edge, Node, ~] = MergeShort(Edge, Node, lt, vt_vec);

% removing resulting loops, sublinks, and twin edges
[Edge, Node] = MainLoop(Edge, Node);

fprintf('-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----\n');

[nc, ~] = hist([Node.connectivity], 1:10);
fprintf('%s|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i|5i\n', ' ML ', ...
        nc(1), ...
        nc(2), ...
        nc(3), ...
        nc(4), ...
        nc(5), ...
        nc(6), ...
        nc(7), ...

```



```

        nc(8), ...
        nc(9), ...
        nc(10), ...
        length(Node) ...
    );

% checking volume after cleanup process
vtotal2 = volumeChecker(Edge);

vol_fraction2 = vtotal2/(dim(1)*dim(2)*dim(3));

fprintf( ...
    '\nPost-cleaned volume fraction of skeleton = %.4f%%\n', ...
    vol_fraction2*100);

HistAll(Edge, Node, dim(3), fname(1:dotInd-1), 'After');

mtit(sprintf('%s - Post Cleanup - LT = %i', fname(1:dotInd-1), lt),...
    'FontSize', 20, ...
    'Color', [0 0 0], ...
    'xoff', 0, 'yoff', .025, ...
    'Interpreter', 'None');

% saving data
if save_switch
    save(sprintf('%s%s_LT%i-Edge.mat', ...
        directory, ...
        fname(1:dotInd-1), ...
        lt ...
    ), ...
    'Edge' ...
    );
    save( ...
        sprintf( ...
            '%s%s_LT%i-Node.mat', ...
            directory, ...
            fname(1:dotInd-1), ...
            lt ...
        ), ...
        'Node' ...
    );
end

% printing data
if print_switch
    print(1, '-dpdf', ...
        sprintf( ...
            '%s%s_LT-%i_pre-cleanup', ...
            directory, ...
            fname(1:end-4), ...
            lt ...
        ) ...
    );
    print(2, '-dpdf', ...
        sprintf( ...
            '%s%s_LT-%i_post-cleanup', ...
            directory, ...
            fname(1:end-4), ...
            lt ...
        ) ...
    );
end

fprintf('\n');
toc;
fprintf('\n');

end

% ----- %

function [Edge, Node] = SkeletonReader(directory, fname)

```

```

% Reads the text file into structures

% fileInd = strfind(fname, 'rec');
fprintf('\nReading %s into structures...', fname);
% disp(sprintf('Read

fid=fopen([directory, fname]);

for i=1:3;
    fgetl(fid);
end %Skip header

% Read file parameters
nvertex=str2num(fscanf(fid,'%s %s %s',[1,1]));
nedge=str2num(fscanf(fid,'%s %s %s',[1,1]));
npoint=str2num(fscanf(fid,'%s %s %s',[1,1]));
iskip = 6;
for i=1:iskip;fgetl(fid);end %Skip transition
GoOn=1; id=0;
while GoOn;
    A=fgetl(fid);
    GoOn=~isempty(A);
    if GoOn;
        id=id+1;
        category=textscan(A,'%s');
        Connect(id).metadata.object=category{1}(1);
        Connect(id).metadata.type=category{1}(3);
        Connect(id).metadata.info=category{1}(4);
        Connect(id).metadata.tag=category{1}(6);
        switch char(Connect(id).metadata.object);
            case 'POINT'
                Connect(id).metadata.ndata=npoint;
            case 'EDGE'
                Connect(id).metadata.ndata=nedge;
            case 'VERTEX'
                Connect(id).metadata.ndata=nvertex;
        end
        nd=str2double(Connect(id).metadata.type{1}(end-1));
        if isnan(nd);
            Connect(id).metadata.ndim=1;
        else
            Connect(id).metadata.ndim=nd;
        end
    end
end

for id=1: numel(Connect)
    GoOn=1;
    while GoOn;
        A=fgetl(fid);
        if ~isempty(A);
            GoOn=~strcmp(A(1),'@');
        end
        [Connect(id).metadata.ndim,Connect(id).metadata.ndata];
        Connect(id).data=
            fscanf(fid,'%g',
                [Connect(id).metadata.ndim,Connect(id).metadata.ndata]);
    end
end
fclose(fid);
Connect(2).data=Connect(2).data+1;

% prepare connections
startedge=cumsum([0,Connect(3).data]);
vtvolume=0; %default volume;
for ie=1:nedge;
    Edge(ie).xdata= ...
        Connect(4).data(1,[startedge(ie)+1:startedge(ie+1)]);
    Edge(ie).ydata= ...
        Connect(4).data(2,[startedge(ie)+1:startedge(ie+1)]);
    Edge(ie).zdata=

```

```

        Connect(4).data(3,[startededge(ie)+1:startededge(ie+1)]);
Edge(ie).linklength=
    (diff(Edge(ie).xdata).^2+
    diff(Edge(ie).ydata).^2+
    diff(Edge(ie).zdata).^2).^(1/2);
Edge(ie).length=
    sum(Edge(ie).linklength);
Edge(ie).endID=
    Connect(2).data(:,ie);
Edge(ie).radius=
    Connect(5).data(startededge(ie)+1:startededge(ie+1));
end
for iv=1:nvertex;
    Node(iv).xdata=Connect(1).data(1,iv);
    Node(iv).ydata=Connect(1).data(2,iv);
    Node(iv).zdata=Connect(1).data(3,iv);
    Node(iv).linkID=
        find((Connect(2).data(2,:)==iv)|(Connect(2).data(1,:)==iv));
    Node(iv).connectivity = numel(Node(iv).linkID);
end

% Storing the initial structures
Network = struct('Edge', {}, ...
    'Node', {}, ...
    'Stage', {});

Network(1).Edge = Edge;
Network(1).Node = Node;
Network(1).Stage = 1;
%
fprintf('Done!\n');

end

% ----- %

function [vtotal] = VolumeChecker(Edge)

vvec = nan(length(Edge), 1);

for ie = 1 : length(Edge)
    [lv, v] = EdgeVolume(Edge(ie), 1);
    vvec(ie) = v;
end

vtotal = sum(vvec);

end

% ----- %

function [Edge, Node] = RemoveDeleted(Edge, Node, ModEdgeID, ModNodeID,
varargin)

cleaner_type = varargin{1};

switch cleaner_type
case 'RemoveLoops'
    edge_delete = ModEdgeID;
    mod_vec = [ModEdgeID; length(Edge) + 1];
    Edge(edge_delete) = []; %translating the edges

    [clinks, tf] = padcat(Node.linkID);
    for ied = 1 : numel(mod_vec) - 1
        %positions of links in clinks matrix
        tp = clinks > mod_vec(ied) & clinks < mod_vec(ied+1);
        lt = clinks(tp) - ied;
        clinks(tp) = lt;
    end

    for in = 1 : size(clinks, 1)
        linkID = clinks(in,:);

```

```

        Node(in).linkID = linkID(tf(in,:));
    end

case 'CollapseLoops'
    E_old = Edge;
    N_old = Node;

    mod_vec = ModEdgeID;
    nmod = numel(mod_vec);
    nn = length(N_old);

    E_new = E_old;
    N_new = N_old;

    for im = 1 : nmod
        edgeID = mod_vec(im);
        endID = E_old(edgeID).endID;
        edge_end_pos = [ ...
            E_old(edgeID).xdata(1), E_old(edgeID).xdata(end); ...
            E_old(edgeID).ydata(1), E_old(edgeID).ydata(end); ...
            E_old(edgeID).zdata(1), E_old(edgeID).zdata(end)];
        node_pos = ...
            [N_old(endID).xdata; ...
            N_old(endID).ydata; ...
            N_old(endID).zdata];
        for ii = 1 : 2
            if node_pos(1) ~= edge_end_pos(1,ii) && ...
                node_pos(2) ~= edge_end_pos(2,ii) && ...
                node_pos(2) ~= edge_end_pos(2,ii)

                new_node_pos = edge_end_pos(:,ii);
                N_new(nn+im).xdata = new_node_pos(1);
                N_new(nn+im).ydata = new_node_pos(2);
                N_new(nn+im).zdata = new_node_pos(3);
                N_new(nn+im).linkID = edgeID;
                N_new(nn+im).connectivity = 1;

                E_new(edgeID).endID = [E_old(edgeID).endID; (nn + im)];
            end
        end
    end

    Edge = E_new;
    Node = N_new;

case {'RemoveSublinks', 'MergeShort', 'RemoveIslandEdges'}
%
    mod_vec = sort([ModEdgeID; length(Edge) + 1]);
    edge_delete = ModEdgeID;

    %translating the edges
    Edge(edge_delete) = [];

    %concatenating the structure elements containing the linkIDs
    [clinks, tf] = padcat(Node.linkID);
    for ied = 1 : numel(mod_vec) - 1
        %positions of links in clinks matrix
        tp = clinks > mod_vec(ied) & clinks < mod_vec(ied+1);
        lt = clinks(tp) - ied;
        clinks(tp) = lt;
    end

    % converting the array of linkID into the Node structure field
    for in = 1 : size(clinks, 1)
        linkID = clinks(in,:);
        Node(in).linkID = linkID(tf(in,:));
    end

    node_trans = sort([ModNodeID; length(Node) + 1]);
    node_delete = ModNodeID;
    Node(node_delete) = [];

```

```

cnodes = [Edge.endID]';
for ind = 1 : numel(node_trans) - 1
    %positions of nodes in cnodes
    tp = cnodes > node_trans(ind) & cnodes < node_trans(ind+1);
    lt = cnodes(tp) - ind;
    cnodes(tp) = lt;
end

% converting the array of linkID into the Edge structure field
for ie = 1 : size(cnodes, 1)
    endID = cnodes(ie,:);
    Edge(ie).endID = endID';
end

case 'RemoveTwins'
    mod_vec = sort([ModEdgeID, length(Edge) + 1]);
    edge_delete = ModEdgeID;

    Edge(edge_delete) = []; % translating the edges

    % updating the linkID entries of the Node structure after the edges
    % were translated in the last loop
    [clinks, tf] = padcat(Node.linkID);
    for ied = 1 : numel(mod_vec) - 1
        %positions of links in clinks matrix
        tp = clinks > mod_vec(ied) & clinks < mod_vec(ied+1);
        lt = clinks(tp) - ied;
        clinks(tp) = lt;
    end

    % converting the array of linkID into the Node structure field
    for in = 1 : size(clinks, 1)
        linkID = clinks(in,:);
        Node(in).linkID = linkID(tf(in,:));
    end

case 'RemoveIslandNodes'
    node_delete = ModNodeID;
    node_trans = sort([ModNodeID, length(Node) + 1]);

    Node(node_delete) = [];

    cnodes = [Edge.endID]';
    for ind = 1 : numel(node_trans) - 1
        %positions of nodes in cnodes
        tp = cnodes > node_trans(ind) & cnodes < node_trans(ind+1);
        lt = cnodes(tp) - ind;
        cnodes(tp) = lt;
    end

    % converting the array of linkID into the Edge structure field
    for ie = 1 : size(cnodes, 1)
        endID = cnodes(ie,:);
        Edge(ie).endID = endID';
    end

case 'RemoveZeroEdge'
    mod_vec = sort([ModEdgeID, length(Edge) + 1]);
    edge_delete = ModEdgeID;

    Edge(edge_delete) = []; % translating the edges
    Node(ModNodeID).linkID(Node(ModNodeID).linkID == ModEdgeID) = [];
    Node(ModNodeID).connectivity = Node(ModNodeID).connectivity - 1;

    % updating the linkID entries of the Node structure after the edges
    % were translated in the last loop
    [clinks, tf] = padcat(Node.linkID);
    for ied = 1 : numel(mod_vec) - 1
        %positions of links in clinks matrix
        tp = clinks > mod_vec(ied) & clinks < mod_vec(ied+1);
        lt = clinks(tp) - ied;
        clinks(tp) = lt;
    end

```

```

        end

        % converting the array of linkID into the Node structure field
        for in = 1 : size(clinks, 1)
            linkID = clinks(in,:);
            Node(in).linkID = linkID(tf(in,:));
        end
    end

end

% ----- %

function HistAll(Edge, Node, dim, varargin)
    varargin_on = isempty(varargin);

    if varargin_on == 0
        initial_switch = ...
            abs(isempty(find(strcmp('Initial', varargin), 1)) - 1);
        after_switch = abs(isempty(find(strcmp('After', varargin), 1)) - 1);
    else
        initial_switch = 0;
        after_switch = 0;
    end

    %% Buffer
    %Declaring a buffer zone so that the nodes with connectivity 1 do not
    %overwhelm the histogram
    bd = 30; %buffer distance
    % Buffer = struct('xlim', {}, 'ylim', {}, 'zlim', {});
    Buffer.xlim = [bd, dim - bd];
    Buffer.ylim = [bd, dim - bd];
    Buffer.zlim = [bd, dim - bd];

    it = 1;
    NodeFit = struct('xdata', {}, ...
        'ydata', {}, ...
        'zdata', {}, ...
        'linkID', {}, ...
        'connectivity', {});

    for in = 1 : length(Node)
        node_position = [Node(in).xdata; Node(in).ydata; Node(in).zdata];
        if node_position(1) > ...
            Buffer.xlim(1) ...
            && node_position(1) ...
            < Buffer.xlim(2) && ...
            node_position(2) > ...
            Buffer.ylim(1) && ...
            node_position(2) < ...
            Buffer.ylim(2) && ...
            node_position(3) > ...
            Buffer.zlim(1) && node_position(3) < Buffer.zlim(2)
            NodeFit(it) = Node(in);
            it = it + 1;
        end
    end

    if initial_switch == 1
        figure(1); clf; hold on;
    end
    if after_switch == 1
        figure(2); clf; hold on;
    end

    CLimit = max([NodeFit.connectivity]);
    % CLimit = 8;
    subplot 221
    c = [NodeFit.connectivity];
    % [n, xout] = hist(c, [1:max(c)]);
    [n, xout] = hist(c, 1:1:CLimit);

```

```

nnorm = n./sum(n);
h1 = bar(xout, nnorm);
set(h1, 'FaceColor', [1 1 1]*.6);
axis([0, max(xout) + 1, 0, max(nnorm) + .1]);
for b = 1 : numel(xout)
    text(xout(b), nnorm(b)+.03, num2str(n(b)), ...
         'FontSize', 12, ...
         'HorizontalAlignment', 'center');
end

xlabel('Coordination #', 'FontSize', 12);
ylabel('Frequency', 'FontSize', 12);
title('Connectivity', 'FontSize', 12);
% xticklabel = get(gca, 'XTickLabel');
% XTickVar = get(gca, 'XTickLabel');
% set(gca, 'XTickLabel', XTickVar*100);

subplot 222
[n, xout] = hist([Edge.radius], 20);
nnorm = n./sum(n);
h2 = bar(xout, nnorm);
set(h2, 'FaceColor', [1 1 1]*.6);
axis([0, max(xout) + 1, 0, max(nnorm) + .1]);

xlabel('Length', 'FontSize', 12);
ylabel('Frequency', 'FontSize', 12);
title('Link Radius', 'FontSize', 12);

subplot 223
[n, xout] = hist([Edge.length], 20);
nnorm = n./sum(n);
h3 = bar(xout, nnorm);
set(h3, 'FaceColor', [1 1 1]*.6);
axis([0, max(xout) + 1, 0, max(nnorm) + .1]);

xlabel('Length', 'FontSize', 12);
ylabel('Frequency', 'FontSize', 12);
title('Link Length', 'FontSize', 12);

subplot 224
[n, xout] = hist(log10([Edge.length]), 20);
nnorm = n./sum(n);
h4 = bar(xout, nnorm);
set(h4, 'FaceColor', [1 1 1]*.6);
axis([0, max(xout) + .1, 0, max(nnorm) + .025]);

xlabel('Length', 'FontSize', 12);
ylabel('Frequency', 'FontSize', 12);
title('Log Link Length', 'FontSize', 12);

end

% ----- %

function [Edge, Node] = IntialInterp(Edge, Node, ninterp)
% Many of the subroutines in this package require that edges be composed of
% less than 4 points. So we add points to edges based on linear
% interpolation.

% ninterp = 4; %all edges will have a minimum of 4 points
fprintf('\nInterpolating edges that contain < %i points...', ninterp);
nptsmod = 0;
for ie = 1 : length(Edge)
    xyz = [Edge(ie).xdata; Edge(ie).ydata; Edge(ie).zdata];
    npoints = size(xyz, 2);
    x.position = xyz;
    x.radius = Edge(ie).radius;
    if npoints < ninterp
        [~, volume0] = Edgevolume(Edge(ie), 1);
        [xi, yi, zi, ri] = EdgeInterp(x, ninterp);
        Edge(ie).xdata = xi;
        Edge(ie).ydata = yi;
    end
end

```

```

        Edge(ie).zdata = zi;
        Edge(ie).linklength = sum(diff([xi; yi; zi], [], 2).^2, 1).^(1/2);
        Edge(ie).length = sum(Edge(ie).linklength);
        Edge(ie).radius = ri;

        nptsmod = nptsmod + 1;
    end
end

fprintf('\n    %i edges were modified\n', nptsmod);

end

% ----- %

function [linkvolume, volume] = EdgeVolume(edge, lscale)

x = [0 cumsum(edge.linklength)]*lscale;
f = pi*(edge.radius*lscale).^2;
vol = .5*diff(x).*(f(1:end-1) + f(2:end));
linkvolume = vol;
volume = sum(vol);

end

% ----- %

function [xi, yi, zi, ri] = EdgeInterp(x, n)

edge_position = [x(end).position(1,:)', ...
    x(end).position(2,:)', ...
    x(end).position(3,:)'];

InterpStruct0.distance = ...
    cat(1, 0, cumsum(sqrt(sum(diff(edge_position, [], 1).^2, 2))));
InterpStruct0.radius = ...
    cat(2, 0, cumsum(sqrt(sum(diff(x(end).radius, [], 2).^2, 1))));
InterpStruct1.position = ...
    interp1(InterpStruct0.distance, edge_position, ...
    linspace(0, InterpStruct0.distance(end), n), 'linear');
InterpStruct1.distance = ...
    cat(1, 0, cumsum(sqrt(sum(diff(InterpStruct1.position, [], 1).^2, 2)...
    )));

e = 1e-4;

InterpStruct1.radius = ...
    interp1(round(InterpStruct0.distance/e)*e, x(end).radius', ...
    round(InterpStruct1.distance/e)*e, 'linear');

xi = InterpStruct1.position(:,1)';
yi = InterpStruct1.position(:,2)';
zi = InterpStruct1.position(:,3)';

ri = InterpStruct1.radius;

end

% ----- %

function [Edge, Node] = RemoveIslands(Edge, Node)
% Removes island nodes and island edges whose end nodes have connectivity
% equal to 1.

% Removing island nodes
cleaner_type = 'RemoveIslandNodes';
island_node = find([Node.connectivity] == 0);
if isempty(island_node) == 0
    [Edge, Node] = RemoveDeleted(Edge, Node, island_node, [], ...
    cleaner_type);
end

```



```

% Removing island edges
cleaner_type = 'RemoveIslandEdges';
island_edge = [];
ModNodeID = [];
for ie = 1 : length(Edge)
    endID = Edge(ie).endID;
    connectivity = [Node(endID).connectivity];
    if isequal(connectivity, [1 1])
        island_edge = [island_edge; ie];
        ModNodeID = [ModNodeID; endID];
    end
end
ModEdgeID = island_edge;

[Edge, Node] = RemoveDeleted(Edge, Node, ModEdgeID, ModNodeID, ...
    cleaner_type);

end

% ----- %

function [Edge, Node, hist_log] = MainLoop(Edge, Node, varargin)
% -----
%
% This is the wrapper script for removing sublinks, twins, and loops.
% Mainloop.m iterates through 'Edge' and 'Node' until all sublinks, twins,
% and loops are removed.
%
% Inputs:
%
%     Edge --> Input 'Edge' structure
%     Node --> Input 'Node' structure
% -----

varargin_on = isempty(varargin);
if varargin_on == 0
    merge_switch = ~isempty(find(strcmp('Merge', varargin), 1));
else
    merge_switch = 0;
end

Connectivity = [Node.connectivity];
[nc, ~] = hist(Connectivity, 1:10);
hist_log(1,:) = nc;

iLim = 1;
dc_sum = 1;

while dc_sum > 0

    iLim = iLim + 1;

    [Edge, Node] = ModifyLoops(Edge, Node, 15);

    if ~merge_switch
        [Edge, Node] = RemoveSublinks(Edge, Node);
    end

    [Edge, Node] = RemoveTwins(Edge, Node);

    Connectivity = [Node.connectivity];
    nc = hist(Connectivity, 1:10);
    hist_log(iLim,:) = nc;

    dc = diff(hist_log(end-1:end,:), [], 1);

    dc_sum = sum(abs(dc));

end

```

```

end

% ----- %

function [Edge, Node] = ModifyLoops(Edge, Node, vthreshold)
%
% -----
% PlotSkeleton.m modifies the input skeleton network by removing loops,
% i.e. edges with only one connected node.
% Inputs:
%
%   'Edge'      --> Input 'Edge' structure
%   'Node'      --> Input 'Node' structure
%   'vthreshold' --> volume threshold for totally removing loop
%
%   'Edge'      --> Output 'Edge' structure
%   'Node'      --> Output 'Node' structure
%
% -----
%

% Removing loops below the threshold length
loopInd = find(diff([Edge.endID]) == 0);
lvolume = nan(numel(loopInd), 1);

for il = 1 : numel(loopInd)
    [~, volume] = EdgeVolume(Edge(loopInd(il)), 1);
    lvolume(il) = volume;
end

% vthreshold = 15; %volume threshold
removeInd = loopInd(lvolume <= vthreshold);
nremove = numel(removeInd);

cleaner_type = 'RemoveLoops';

ModNodeID = [];

ModEdgeID = nan(floor(length(Node)*.2), 1);
ime = 1;

for il = 1 : nremove
    mod_node = Edge(removeInd(il)).endID(1);
    iedge_delete = removeInd(il);

    ModEdgeID(ime) = iedge_delete;
    ime = ime + 1;

    % removing the loop edge from the linkID entry of the Node
    % structure
    Node(mod_node).linkID = ...
        Node(mod_node).linkID(Node(mod_node).linkID ~= removeInd(il));
    Node(mod_node).connectivity = Node(mod_node).connectivity - 1;
end

ModEdgeID(isnan(ModEdgeID)) = [];

[Edge, Node] = RemoveDeleted( ...
    Edge, Node, ModEdgeID, ModNodeID, cleaner_type);

% Collapsing loops into single edges
loopInd = find(diff([Edge.endID]) == 0);
lvolume = nan(numel(loopInd), 1);

for il = 1 : numel(loopInd)
    [~, volume] = EdgeVolume(Edge(loopInd(il)), 1);
    lvolume(il) = volume;
end

```

```

collapseInd = loopInd(lvolume > vthreshold);
ncollapse = numel(collapseInd);

for il = 1 : ncollapse
%    disp(il);

    edgeID = collapseInd(il);
    nodeID = unique(Edge(edgeID).endID);

    link_pos = ...
        [Edge(edgeID).xdata; Edge(edgeID).ydata; Edge(edgeID).zdata];
    [~, loop_volume] = EdgeVolume(Edge(edgeID), 1);

    uE = 1e-6;

    nUniqueLinkPos = ...
        size(unique(round(link_pos(:,2:end-1)'./uE)*uE, 'rows'), 2);

    if nUniqueLinkPos == 1 % adhoc modification for Maddy's research

        fprintf('\nwarning: Found linear loops at edge %i\n\n', edgeID);

        Edge(edgeID).xdata = Edge(edgeID).xdata(1:2);
        Edge(edgeID).ydata = Edge(edgeID).ydata(1:2);
        Edge(edgeID).zdata = Edge(edgeID).zdata(1:2);
        Edge(edgeID).linklength = Edge(edgeID).linklength(1);
        Edge(edgeID).length = Edge(edgeID).linklength(1);
        Edge(edgeID).radius = Edge(edgeID).radius(2:end-1);

        nn = length(Node);
        edge_position = [Edge(edgeID).xdata; ...
            Edge(edgeID).ydata; ...
            Edge(edgeID).zdata];
        node_position = [Node(nodeID).xdata; ...
            Node(nodeID).ydata; ...
            Node(nodeID).zdata];
        node_position = node_position(:, ones(1,size(edge_position, 2)));

        e = 1e-3;

        ipos = find(sum(edge_position <= node_position + e & ...
            edge_position >= node_position - e) == 3);

        if ipos == 1
            nnode_position = edge_position(:,end);
        else
            nnode_position = edge_position(:,1);
        end

        Node(nn+1).xdata = nnode_position(1);
        Node(nn+1).ydata = nnode_position(2);
        Node(nn+1).zdata = nnode_position(3);
        Node(nn+1).linkID = edgeID;
        Node(nn+1).connectivity = numel(Node(nn+1).linkID);

        Edge(edgeID).endID = [nodeID; nn + 1];

    else

        % finding the index of the value that is half the distance along
        % the loop
        halfway = Edge(edgeID).length/2;
        edge_dist_vec = cumsum(Edge(edgeID).linklength);
        imax = find(diff(sign(edge_dist_vec - halfway)));
        if imax == 1 % adhoc modification for Maddy's research
            imax = imax + 1;
        end
        npts_tot = size(link_pos, 2); % # of points that make up the edge

        % splitting the original edge into 2 edges, essentially turning the
        % loop into a twin

```

```

npts1 = imax - 1; % # of points in the 1st new edge

% position data of the 1st new edge
edge(1).position = link_pos(:,1:npts1);
edge(1).radius = Edge(edgeID).radius(1:npts1);

% position data of the 2nd new edge
edge(2).position = fliplr(link_pos(:,npts1:npts_tot));
edge(2).radius = fliplr(Edge(edgeID).radius(npts1:npts_tot));

[~, minptID] = min([size(edge(1).position, 2), ...
    size(edge(2).position, 2)]);
[cmaxpt, maxptID] = max([size(edge(1).position, 2), ...
    size(edge(2).position, 2)]);

% this will happen if the two new edges coincidentally have
% the same number of points
if minptID == maxptID
    minptID = 1;
    maxptID = 2;
    cmaxpt = size(edge(maxptID).position, 2);
end

[xmin, ymin, zmin, rmin] = EdgeInterp(edge, cmaxpt);

LoopInterp = struct('xdata', {}, ...
    'ydata', {}, ...
    'zdata', {}, ...
    'radius', {});

LoopInterp(minptID).xdata = xmin;
LoopInterp(minptID).ydata = ymin;
LoopInterp(minptID).zdata = zmin;
LoopInterp(minptID).radius = rmin;

LoopInterp(maxptID).xdata = edge(maxptID).position(1,:);
LoopInterp(maxptID).ydata = edge(maxptID).position(2,:);
LoopInterp(maxptID).zdata = edge(maxptID).position(3,:);
LoopInterp(maxptID).radius = edge(maxptID).radius;

xd = cat(1, LoopInterp(1).xdata, LoopInterp(2).xdata);
yd = cat(1, LoopInterp(1).ydata, LoopInterp(2).ydata);
zd = cat(1, LoopInterp(1).zdata, LoopInterp(2).zdata);
rd = cat(1, LoopInterp(1).radius, LoopInterp(2).radius);

ad = pi*rd.^2;
a = sum(ad);

xi = sum(xd.*ad)./a; %area-weighted average;
yi = sum(yd.*ad)./a; %area-weighted average;
zi = sum(zd.*ad)./a; %area-weighted average;

ri = sqrt(a/pi); %area-weighted average

% smoothing the new edge
xs = smooth(xi', .3);
ys = smooth(yi', .3);
zs = smooth(zi', .3);

% calculating the link lengths
linklengthi = sum(diff([xs'; ys'; zs'], [], 2).^2, 1).^(1/2);

% calculating the length of the new edge
lengthi = sum(linklengthi);

% updating the Edge structure
Edge(edgeID).xdata = xs';
Edge(edgeID).ydata = ys';
Edge(edgeID).zdata = zs';
Edge(edgeID).linklength = linklengthi;
Edge(edgeID).length = lengthi;

```

```

nn = length(Node);
edge_position = [Edge(edgeID).xdata; ...
    Edge(edgeID).ydata; ...
    Edge(edgeID).zdata];
node_position = [Node(nodeID).xdata; ...
    Node(nodeID).ydata; ...
    Node(nodeID).zdata];
node_position = node_position(:, ones(1,size(edge_position, 2)));

e = 1e-3;

ipos = find(sum(edge_position<=node_position+e & ...
    edge_position>=node_position-e) == 3);

if ipos == 1
    nnode_position = edge_position(:,end);
else
    nnode_position = edge_position(:,1);
end

Node(nn+1).xdata = nnode_position(1);
Node(nn+1).ydata = nnode_position(2);
Node(nn+1).zdata = nnode_position(3);
Node(nn+1).linkID = edgeID;
Node(nn+1).connectivity = numel(Node(nn+1).linkID);

Edge(edgeID).endID = [nodeID; nn + 1];
Edge(edgeID).radius = ri;

% calculating the volume of the new edge
[~, volume] = Edgevolume(Edge(edgeID), 1);

% growing the links to conserve the volume of the original twins
% added together
vrat = loop_volume/volume;
Edge(edgeID).radius = Edge(edgeID).radius.*sqrt(vrat);

end

end

end

% ----- %

function [Edge, Node] = RemoveSublinks(Edge, Node)

cleaner_type = 'RemoveSublinks';

% narrowing the number of nodes to the ones with coordination # of 2
coord2 = find([Node.connectivity] == 2);
nc2 = numel(coord2);

for ic2 = 1 : nc2
    noi = coord2(ic2);
    eoi = Node(noi).linkID;
    endID = [Edge(eoi).endID];
    opp_ends = endID(endID ~= coord2(ic2))';
    unique_nodes = unique(opp_ends);
    if numel(unique_nodes) > 1 %erogo there is a sublink
        iedge_keep = min(eoi);
        iedge_delete = eoi(eoi ~= iedge_keep);
        inode_delete = noi;

        % position matrices of the first and second connecting edges
        e1p = [Edge(eoi(1)).xdata; ...
            Edge(eoi(1)).ydata; ...
            Edge(eoi(1)).zdata];
        e2p = [Edge(eoi(2)).xdata; ...
            Edge(eoi(2)).ydata; ...
            Edge(eoi(2)).zdata];
    end
end

```

```

% radius of the first and second connecting edges
e1r = Edge(eoi(1)).radius;
e2r = Edge(eoi(2)).radius;

% determining the index and flipping order
[flip_switch, iflip, oflip] = FlipSwitch(e1p, e2p, cleaner_type);

MultiEdge = struct('position', {}, 'radius', {});

MultiEdge(1).position = [e1p(1,:); e1p(2,:); e1p(3,:)];
MultiEdge(2).position = [e2p(1,:); e2p(2,:); e2p(3,:)];
MultiEdge(1).radius = e1r;
MultiEdge(2).radius = e2r;

presort = struct('position', {}, 'radius', {});
postsort = struct('position', {}, 'radius', {});

switch flip_switch
    case 1
        presort(oflip(oflip ~= iflip)).position = ...
            MultiEdge(oflip(oflip~=iflip)).position;
        presort(iflip).position = ...
            flipplr(MultiEdge(iflip).position);
        presort(oflip(oflip ~= iflip)).radius = ...
            MultiEdge(oflip(oflip ~= iflip)).radius;
        presort(iflip).radius = ...
            flipplr(MultiEdge(iflip).radius);

        postsort(1).position = presort(oflip(1)).position;
        postsort(2).position = presort(oflip(2)).position;
        postsort(1).radius = presort(oflip(1)).radius;
        postsort(2).radius = presort(oflip(2)).radius;

    case 0
        postsort(oflip(1)).position = MultiEdge(1).position;
        postsort(oflip(2)).position = MultiEdge(2).position;
        postsort(oflip(1)).radius = MultiEdge(1).radius;
        postsort(oflip(2)).radius = MultiEdge(2).radius;
end

postsort(2).position(:,1) = [];
postsort(2).radius(1) = [];

% appended positions of the edge
append_position = [[postsort(1).position], [postsort(2).position]];
append_radius = [[postsort(1).radius], [postsort(2).radius]];

% positions of the nodes
node_positions = ...
    [Node(opp_ends(1)).xdata, Node(opp_ends(2)).xdata; ...
     Node(opp_ends(1)).ydata, Node(opp_ends(2)).ydata; ...
     Node(opp_ends(1)).zdata, Node(opp_ends(2)).zdata];

% flipping the edge if neccessary
[append_position, append_radius] = ...
    FlipEdge(append_position, node_positions, ...
    'Radius', append_radius);

% updating the edge structure
Edge(iedge_keep).xdata = append_position(1,:);
Edge(iedge_keep).ydata = append_position(2,:);
Edge(iedge_keep).zdata = append_position(3,:);
Edge(iedge_keep).linklength = ...
    sum(diff(append_position, [], 2).^2).^(1/2);
Edge(iedge_keep).length = sum(Edge(iedge_keep).linklength);
Edge(iedge_keep).endID = opp_ends';
Edge(iedge_keep).radius = append_radius;

other_node = ...
    Edge(iedge_delete).endID(Edge(iedge_delete).endID ~=
noi);
old_loc = Node(other_node).linkID == iedge_delete;

```

```

Node(other_node).linkID(old_loc) = iedge_keep;

[Edge, Node] = ...
    RemoveDeleted(...
        Edge, Node, iedge_delete, inode_delete, cleaner_type);

coord2(ic2+1:end) = coord2(ic2+1:end) - 1;

end
end
end

% ----- %

function [Edge, Node] = RemoveTwins(Edge, Node)

cleaner_type = 'RemoveTwins';

E = Edge;
N = Node;
ne = length(Edge);
nn = length(Node);

idelete_edge = nan(1, floor(ne*.2));
idelete_node = nan(1, floor(nn*.2));

ide = 1;

for in = 1 : nn
    %IDs of the egdes that are connected to Node in
    linkIDs = N(in).linkID;

    %IDs of the nodes that are connected to the connecting edges [in;
    %new_node]
    endID = [E(linkIDs).endID];

    %node IDs that ~= in but are connected to the connecting edges
    %[new_nodes]
    cnodes = endID(endID ~= in)';
    unique_nodes = UniqueVal(cnodes);

    %ergo there is a twin present
    if numel(unique_nodes) < numel(cnodes)
        if numel(unique_nodes) == 1
            nrep = numel(cnodes);
        else
            nrep = hist(cnodes, unique_nodes);
        end
        rep_nodes = unique_nodes(nrep > 1);
        for ir = 1 : numel(rep_nodes) %loops through repeating indicies
            %list of nodes that are shared by the twins
            noi = [in; rep_nodes(ir)];
            %list of edges that comprise the twins
            eoi = linkIDs(cnodes == rep_nodes(ir));
            twin = struct('xdata', {}, ...
                'ydata', {}, ...
                'zdata', {}, ...
                'radius', {}, ...
                'endID', {});
            npts = nan(numel(eoi), 1);
            for it = 1 : numel(eoi) %looping through the twin edges
                if isequal(E(eoi(it)).endID, noi) == 1 || ...
                    isequal(flipud(E(eoi(it)).endID), noi) == 1
                    twin(it).xdata = E(eoi(it)).xdata;
                    twin(it).ydata = E(eoi(it)).ydata;
                    twin(it).zdata = E(eoi(it)).zdata;
                    twin(it).radius = E(eoi(it)).radius;
                    twin(it).linklength = E(eoi(it)).linklength;
                    twin(it).length = E(eoi(it)).length;
                    twin(it).endID = E(eoi(it)).endID;
                    twin(it).radius = E(eoi(it)).radius;
                end
            end
        end
    end
end

```

```

        npts(it) = numel(twin(it).xdata);
        [~, tv] = EdgeVolume(twin(it), 1);
        twin(it).volume = tv;
    end
end

% in case the twins have the same number of points
[tmax, tmaxInd] = max(npts);

index_vec = 1:length(twin);

other_index = index_vec(index_vec ~= tmaxInd);

e1p = [twin(tmaxInd).xdata; ...
       twin(tmaxInd).ydata; ...
       twin(tmaxInd).zdata];

e1r = twin(tmaxInd).radius;

xd = nan(numel(npts), tmax);
yd = nan(numel(npts), tmax);
zd = nan(numel(npts), tmax);
rd = nan(numel(npts), tmax);

xd(1,:) = e1p(1,:);
yd(1,:) = e1p(2,:);
zd(1,:) = e1p(3,:);

rd(1,:) = e1r;

for it = 1 : length(other_index)
    e2p = [twin(other_index(it)).xdata; ...
           twin(other_index(it)).ydata; ...
           twin(other_index(it)).zdata];

    e2r = twin(other_index(it)).radius;

    [flip_switch, ~, ~] = FlipSwitch(e1p, e2p, cleaner_type);

    switch flip_switch
        case 0
            MultiEdge(1).position = ...
                [e1p(1,:); e1p(2,:); e1p(3,:)];
            MultiEdge(2).position = ...
                [e2p(1,:); e2p(2,:); e2p(3,:)];

            MultiEdge(1).radius = e1r;
            MultiEdge(2).radius = e2r;
        case 1
            MultiEdge(1).position = e1p;
            MultiEdge(2).position = fliplr(e2p);

            MultiEdge(1).radius = e1r;
            MultiEdge(2).radius = fliplr(e2r);
    end

    [xi, yi, zi, ri] = EdgeInterp(MultiEdge, tmax);

    xd(it+1,:) = xi;
    yd(it+1,:) = yi;
    zd(it+1,:) = zi;

    rd(it+1,:) = ri;
end

% calculating the cross-sectional area at each point in the
% edge
ad = pi*rd.^2;
a = sum(ad);

```



```

xa = sum(xd.*ad)./a; %area-weighted average;
ya = sum(yd.*ad)./a; %area-weighted average;
za = sum(zd.*ad)./a; %area-weighted average;

ri = nan(1, size(ad, 2));
ri(1) = rd(1,1);
ri(end) = rd(1,end);

ri(2:end-1) = sqrt(a(2:end-1)/pi); %area-weighted average

% smoothing the new edge
xs = smooth(xa', .3);
ys = smooth(ya', .3);
zs = smooth(za', .3);

xs(1) = xi(1); xs(end) = xi(end);
ys(1) = yi(1); ys(end) = yi(end);
zs(1) = zi(1); zs(end) = zi(end);

position_check = [xs'; ys'; zs'];

node_positions = [Node(noi(1)).xdata, Node(noi(2)).xdata; ...
    Node(noi(1)).ydata, Node(noi(2)).ydata; ...
    Node(noi(1)).zdata, Node(noi(2)).zdata];

% flipping the edge if neccessary
[position_check, ri] = ...
    FlipEdge(...
    position_check, node_positions, 'Radius', ri);

% calculating the link lengths
linklengthi = sum(diff(position_check, [], 2).^2, 1).^(1/2);

% calculating the length of the new edge
lengthi = sum(linklengthi);

% storing the edge ID to be kept
% isave_edge = eoi(tmaxInd);
isave_edge = min(eoi);

% storing the new values in the Edge structure
E(isave_edge).xdata = position_check(1,:);
E(isave_edge).ydata = position_check(2,:);
E(isave_edge).zdata = position_check(3,:);
E(isave_edge).linklength = linklengthi;
E(isave_edge).length = lengthi;
E(isave_edge).endID = noi;
E(isave_edge).radius = ri;

% calculating the volume of the edge
[~, volume] = EdgeVolume(E(isave_edge), 1);

% growing the links to conserve the volume of the
% original twins added together
vrat = sum([twin.volume])/volume;
E(isave_edge).radius = E(isave_edge).radius.*sqrt(vrat);

% storing the IDs of the edges to be deleted
EdeleteID = eoi(eoi ~= isave_edge);
ndt = numel(EdeleteID);
idelete_edge(ide:(ide+ndt-1)) = EdeleteID;

ide = ide + ndt;

% updating the Node structure
%updating the linkIDs
N(noi(1)).linkID = ...
    N(noi(1)).linkID(~ismember( ...
    N(noi(1)).linkID, eoi(eoi ~=isave_edge)));
%updating the connectivity
N(noi(1)).connectivity = ...
    N(noi(1)).connectivity - (length(twin) - 1);

```

```

        N(noi(2)).linkID = ...
            N(noi(2)).linkID(~ismember( ...
                N(noi(2)).linkID, eoi(eoi ~=isave_edge)));
        N(noi(2)).connectivity = ...
            N(noi(2)).connectivity - (length(twin) - 1);
    end
end

end

idelete_edge = idelete_edge(~isnan(idelete_edge));

[Edge, Node] = RemoveDeleted(E, N, idelete_edge, idelete_node, ...
    cleaner_type);

end

% ----- %

function [Edge, Node, vt_vec] = MergeShort(Edge, Node, lt, vt_vec)
% -----
%
% Removes edges shorter than 'lt' by merging connected edges. Total volume
% is preserved in this process.
% -----

cleaner_type = 'MergeShort';

shortID = find([Edge.length] < lt, 1); %ID of edge that could be too short
GoShort = 1;
it = 0;

while GoShort > 0

    it = it + 1;

    % In case, the shortID is an island edge
    IE_endID = Edge(shortID).endID;
    IE_connectivity = [Node(IE_endID).connectivity];
    if isequal(IE_connectivity, [1 1])
        [Edge, Node] = RemoveDeleted(Edge, Node, shortID, IE_endID,
'RemoveIslandEdges');
    else
        cnodes = [Edge(shortID).endID];
        inode_keep = min(cnodes);
        inode_delete = cnodes(cnodes ~= inode_keep);
        iedge_delete = shortID;
        clinks = padcat(Node(cnodes).linkID);

        % finding the point on the connecting edges corresponding to a
length
        % "BoundStruct(1).length" or "right_length" away from the node. This
will be the
        % point that will be connected to the straight line connecting the
COM
        % of the short edge to the connecting edge.
        EdgeUpdate = struct('xdata', {}, ...
            'ydata', {}, ...
            'zdata', {}, ...
            'linklength', {}, ...
            'length', {}, ...
            'endID', {}, ...
            'radius', {});

        connectivity = [Node(cnodes).connectivity];
        offshoot = cnodes(connectivity == 1);

        if ~isempty(offshoot) %pruning short offshoots

```



```

midpts = straight_pos(:,1:end-1)+diff(straight_pos, [], 2)/2;
vw = midpts.*repmat(vlinks_mod, 3, 1);
COM = sum(vw, 2)./v_mod;

between_vec = nan(1, 3);

% locating the index where I want to split the straight segment
between_vec(1) = inbetween(straight_pos(1,:), COM(1));
between_vec(2) = inbetween(straight_pos(2,:), COM(2));
between_vec(3) = inbetween(straight_pos(3,:), COM(3));

inan = isnan(between_vec);
if ~isempty(find(inan, 1))
    between_vec(inan) = unique(between_vec(~inan));
end

iCOM = unique(between_vec);
if iCOM == 1
    iCOM = 2;
end
if iCOM == size(straight_pos, 2);
    iCOM = size(straight_pos, 2) - 1;
end

% If the straight edge only contains 3 coordinates, the COM is
% automatically assigned to be the median of the points.
if size(straight_pos, 2) == 3
    iCOM = 2;
end

new_node_pos = straight_pos(:,iCOM);

connecting % finding the distance from the COM to the left and right
% nodes
left_node_pos = [Node(cnodes(1)).xdata; ...
    Node(cnodes(1)).ydata; ...
    Node(cnodes(1)).zdata];
right_node_pos = [Node(cnodes(2)).xdata; ...
    Node(cnodes(2)).ydata; ...
    Node(cnodes(2)).zdata];

BoundStruct = struct('length', {}, ...
    'pts', {}, ...
    'radius', {});

% 1 --> left bound; 2 --> right bound
BoundStruct(1).length = sum(diff([COM, left_node_pos], [],
2).^2, 1).^(1/2);
BoundStruct(2).length = sum(diff([COM, right_node_pos], [],
2).^2, 1).^(1/2);

%the center of mass pt is shared
BoundStruct(1).pts = straight_pos(:,1:iCOM);
BoundStruct(2).pts = straight_pos(:,iCOM:end);

BoundStruct(1).radius = Edge(shortID).radius(1:iCOM);
BoundStruct(2).radius = Edge(shortID).radius(iCOM:end);

side of % calculating the volumes of the individual segments on either
% COM
left_volume = sum(vlinks_mod(1:iCOM));
right_volume = sum(vlinks_mod(iCOM+1:end));

BoundStruct(1).volume = left_volume;
BoundStruct(2).volume = right_volume;

for ii = 1 : 2
    noi = cnodes(ii);
    node_pos = [Node(noi).xdata; Node(noi).ydata;
Node(noi).zdata];

```

```

        clinks1 = clinks(ii,:);
        clinks1 = clinks1(clinks1 ~= shortID);
        nclinks1 = numel(clinks1(isnan(clinks1) ~= 1));
        vdiv = BoundStruct(ii).volume/nclinks1;
        for il = 1 : numel(clinks1(isnan(clinks1) ~= 1))
            edge_pos = [Edge(clinks1(il)).xdata; ...
                        Edge(clinks1(il)).ydata; ...
                        Edge(clinks1(il)).zdata];
            endInd = find(sum([round(edge_pos(1,:)*1e4)/1e4 ==
round(node_pos(1)*1e4)/1e4; ...
round(edge_pos(2,:)*1e4)/1e4 ==
round(node_pos(2)*1e4)/1e4; ...
round(edge_pos(3,:)*1e4)/1e4 ==
round(node_pos(3)*1e4)/1e4], 1) == 3, 1);
            if endInd == size(edge_pos, 2)
                edge_pos = fliplr(edge_pos); %needs to be flipped
                lradius = fliplr(Edge(clinks1(il)).radius);
            else
                lradius = Edge(clinks1(il)).radius;
            end

            edge_dist_vec = [0 cumsum(sum(diff(edge_pos, [],
2).^2).^((1/2)))];
            if edge_dist_vec(end) > BoundStruct(ii).length
                [~, imin] = min(abs(edge_dist_vec -
BoundStruct(ii).length));
                if imin >= numel(edge_dist_vec)
                    iattach = numel(edge_dist_vec) - 1;
                else
                    iattach = imin;
                end
            else
                iattach = numel(edge_dist_vec) - 1; %for connecting
edges that are shorter than the shortIDs
            end

            % Creating temporary edge that appends the appropriate
side
            % of the short-side with the connecting nodes. This edge
            % will be called etemp1
            e1p = BoundStruct(ii).pts;
            e2p = [edge_pos(1,1:iattach); ...
                    edge_pos(2,1:iattach); ...
                    edge_pos(3,1:iattach)];
            e1r = BoundStruct(ii).radius;
            e2r = lradius(1:iattach);

            [flip_switch, iflip, oflip] = FlipSwitch(e1p, e2p,
cleaner_type);

            MultiEdge = struct('position', {}, 'radius', {});

            MultiEdge(1).position = e1p;
            MultiEdge(2).position = e2p;

            MultiEdge(1).radius = e1r;
            MultiEdge(2).radius = e2r;

            presort = struct('position', {}, 'radius', {});
            postsort = struct('position', {}, 'radius', {});

            append_position = [];
            append_radius = [];

            switch flip_switch
                case 1
                    presort(oflip(oflip ~= iflip)).position =
MultiEdge(oflip(oflip ~= iflip)).position;
                    presort(iflip).position =
fliplr(MultiEdge(iflip).position);
                    presort(oflip(oflip ~= iflip)).radius =
MultiEdge(oflip(oflip ~= iflip)).radius;

```



```

        MultiEdge(1).position = temp_pos1;
        MultiEdge(2).position = fliplr(temp_pos2);

        MultiEdge(1).radius = append_radius;
        MultiEdge(2).radius = fliplr(append_radius);
    end

    nme = length(MultiEdge);

    xd = nan(nme, size(MultiEdge(1).position, 2));
    yd = nan(nme, size(MultiEdge(1).position, 2));
    zd = nan(nme, size(MultiEdge(1).position, 2));
    rd = nan(nme, size(MultiEdge(1).position, 2));

    for ime = 1 : nme
        xd(ime,:) = MultiEdge(ime).position(1,:);
        yd(ime,:) = MultiEdge(ime).position(2,:);
        zd(ime,:) = MultiEdge(ime).position(3,:);
        rd(ime,:) = MultiEdge(ime).radius;
    end

    % Averaging the temporary edges to make one edge
    ad = pi*rd.^2;
    a = sum(ad);

    xa = sum(xd.*ad)./a; %area-weighted average;
    ya = sum(yd.*ad)./a; %area-weighted average;
    za = sum(zd.*ad)./a; %area-weighted average;

    xa(1) = xd(1,1); xa(end) = xd(1,end);
    ya(1) = yd(1,1); ya(end) = yd(1,end);
    za(1) = zd(1,1); za(end) = zd(1,end);

    %          ri = sqrt(a/pi); %area-weighted average
    ri = append_radius;

    % concatenating the position vectors
    npos = [xa; ya; za];

    % appending the data from the first temp edge above the
iattach
    % index
    edge_extra = edge_pos(:,iattach:end);
    radius_extra = radius(:,iattach:end);

    [flip_switch, iflip, oflip] = FlipSwitch(npos,
edge_extra, 'MergeShort');

    MultiEdge = struct('position', {}, 'radius', {});

    MultiEdge(1).position = npos;
    MultiEdge(2).position = edge_extra;

    MultiEdge(1).radius = ri;
    MultiEdge(2).radius = radius_extra;

    presort = struct('position', {}, 'radius', {});
    postsort = struct('position', {}, 'radius', {});

    append_position = [];
    append_radius = [];

    switch flip_switch
        case 1
            presort(oflip(oflip ~= iflip)).position =
MultiEdge(oflip(oflip~=iflip)).position;
            presort(iflip).position =
fliplr(MultiEdge(iflip).position);
            presort(oflip(oflip ~= iflip)).radius =
MultiEdge(oflip(oflip ~= iflip)).radius;
            presort(iflip).radius =
fliplr(MultiEdge(iflip).radius);

```

```

presort(oflip(1)).position; postsort(1).position =
presort(oflip(2)).position; postsort(2).position =
                                postsort(1).radius = presort(oflip(1)).radius;
                                postsort(2).radius = presort(oflip(2)).radius;

                                postsort(2).position(:,1) = [];
                                postsort(2).radius(1) = [];

                                append_position = [[postsort(1).position],
[postsort(2).position]];                                append_radius = [[postsort(1).radius],
[postsort(2).radius]];

                                % flipping the edge id neccessary
                                [append_position, append_radius] =
FlipEdge(append_position, new_node_pos, ...
                                'Radius', append_radius);

                                case 0
MultiEdge(1).position;                                postsort(oflip(1)).position =
MultiEdge(2).position;                                postsort(oflip(2)).position =
                                postsort(oflip(1)).radius = MultiEdge(1).radius;
                                postsort(oflip(2)).radius = MultiEdge(2).radius;

                                postsort(2).position(:,1) = [];
                                postsort(2).radius(1) = [];

                                append_position = [[postsort(1).position],
[postsort(2).position]];                                append_radius = [[postsort(1).radius],
[postsort(2).radius]];

                                % flipping the edge id neccessary
                                [append_position, append_radius] =
FlipEdge(append_position, new_node_pos, ...
                                'Radius', append_radius);
                                end

                                new_endID = Edge(clinks1(il)).endID;
                                inew_endID = Edge(clinks1(il)).endID == noi;
                                new_endID(inew_endID) = inode_keep;

                                leu = length(EdgeUpdate) + 1;

                                EdgeUpdate(leu).xdata = append_position(1,:);
                                EdgeUpdate(leu).ydata = append_position(2,:);
                                EdgeUpdate(leu).zdata = append_position(3,:);
                                EdgeUpdate(leu).linklength = sum((diff(append_position,
[], 2).^2), 1).^(1/2);
                                EdgeUpdate(leu).length =
sum(EdgeUpdate(leu).linklength);
                                EdgeUpdate(leu).endID = new_endID;
                                EdgeUpdate(leu).radius = append_radius;

                                [~, volume0] = Edgevolume(EdgeUpdate(leu), 1);

                                [~, original_clink_v] = Edgevolume(Edge(clinks1(il)),
1);

                                new_volume = original_clink_v + vdiv;
                                vrat = new_volume/volume0;

                                EdgeUpdate(leu).radius =
                                EdgeUpdate(leu).radius*sqrt(vrat);
                                end
                                end

```



```

end

clinks_sorted = [clinks(1,:)'; clinks(2,:)'];

u1 = clinks_sorted(~isnan(clinks_sorted));
u1 = u1(u1 ~= shortID);

Node(inode_keep).xdata = new_node_pos(1);
Node(inode_keep).ydata = new_node_pos(2);
Node(inode_keep).zdata = new_node_pos(3);
Node(inode_keep).linkID = u1';
Node(inode_keep).connectivity = numel(u1);

% creating a temporary straight edge out of the short edge
if straighten_switch == 1
    for iup = 1 : length(EdgeUpdate)
        final_position = [EdgeUpdate(iup).xdata; ...
            EdgeUpdate(iup).ydata; ...
            EdgeUpdate(iup).zdata];
        final_radius = EdgeUpdate(iup).radius;
        node_order = EdgeUpdate(iup).endID == inode_keep;
        if isequal(node_order, [0; 1]) == 1
            EdgeUpdate(iup).endID = flipud(EdgeUpdate(iup).endID);
        end
        Edge(u1(iup)).xdata = final_position(1,:);
        Edge(u1(iup)).ydata = final_position(2,:);
        Edge(u1(iup)).zdata = final_position(3,:);
        Edge(u1(iup)).linklength = sum(diff(final_position, [],
2).^2, 1).^(1/2);
        Edge(u1(iup)).length = sum(Edge(u1(iup)).linklength);
        Edge(u1(iup)).endID = EdgeUpdate(iup).endID;
        Edge(u1(iup)).radius = final_radius;
    end
else
    for iup = 1 : length(EdgeUpdate)
        final_position = [EdgeUpdate(iup).xdata; ...
            EdgeUpdate(iup).ydata; ...
            EdgeUpdate(iup).zdata];
        final_radius = EdgeUpdate(iup).radius;
        endID = EdgeUpdate(iup).endID;
        v = [Node(endID(1)).xdata, Node(endID(2)).xdata; ...
            Node(endID(1)).ydata, Node(endID(2)).ydata; ...
            Node(endID(1)).zdata, Node(endID(2)).zdata];

        final_position = FlipEdge(final_position, v);

        Edge(u1(iup)).xdata = final_position(1,:);
        Edge(u1(iup)).ydata = final_position(2,:);
        Edge(u1(iup)).zdata = final_position(3,:);
        Edge(u1(iup)).linklength = sum(diff(final_position, [],
2).^2, 1).^(1/2);
        Edge(u1(iup)).length = sum(Edge(u1(iup)).linklength);
        Edge(u1(iup)).endID = EdgeUpdate(iup).endID;
        Edge(u1(iup)).radius = final_radius;
    end
end

[Edge, Node] = RemoveDeleted(Edge, Node, iedge_delete, inode_delete,
cleaner_type); %removing the deleted nodes and edges

% checkin gto make sure that the edge is not an island
if Node(inode_keep).connectivity == 1 &&
Node(Edge(Node(inode_keep).linkID).endID(Edge(Node(inode_keep).linkID).endID
~= inode_keep)).connectivity == 1

    [Edge, Node] = RemoveDeleted(Edge, Node, iedge_delete,
inode_delete, 'RemoveIslandEdges'); %removing the deleted nodes and edges

else

    %%%%%%%%%%%
    % Identifying the edges and nodes immediately surrounding the short

```

```

% edge for speed.
cn_lv11 = unique([Edge(Node(inode_keep).linkID).endID]);
cn_lv11 = cn_lv11(cn_lv11 ~= inode_keep);

ce_lv11 = Node(inode_keep).linkID';

ce_lv12 = unique([Node(cn_lv11).linkID]');
ce_lv12_loc = ~ismember(ce_lv12, ce_lv11);
ce_lv12 = ce_lv12(logical(ce_lv12_loc));

cn_lv12 = unique([Edge(ce_lv12).endID]);
cn_lv12_loc = ~ismember(cn_lv12, cn_lv11);
cn_lv12 = cn_lv12(logical(cn_lv12_loc));

target_nodes = [inode_keep; cn_lv11; cn_lv12];
target_edges = [ce_lv11; ce_lv12];

% truncating the edges and nodes structures for speed
TargetNode = Node(target_nodes);
TargetEdge = Edge(target_edges);

for itn = 1 : numel(target_nodes)
    TargetNode(itn).GlobalNodeID = target_nodes(itn);
    TargetNode(itn).GlobalLinkID = TargetNode(itn).linkID;
    TargetNode(itn).GlobalConnectivity =
numel(TargetNode(itn).linkID);
end

for ite = 1 : numel(target_edges)
    TargetEdge(ite).GlobalEdgeID = target_edges(ite);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Translating to their local indicies
GlobalEdgeConnect = [TargetEdge.endID]'; %rows are edges, and
columns are the indicies of the endID's
LocalEdgeConnect = nan(size(GlobalEdgeConnect));
unique_nodes = unique(GlobalEdgeConnect);
for iun = 1 : numel(unique_nodes)
    loc = GlobalEdgeConnect == target_nodes(iun);
    [r, ~] = find(sum(loc, 2));
    LocalEdgeConnect(loc) = iun;
    TargetNode(iun).linkID = sort(r)';
    TargetNode(iun).connectivity = numel(r);
    TargetNode(iun).nodeID = iun;
end

for ilocal = 1 : numel(target_edges)
    TargetEdge(ilocal).endID = LocalEdgeConnect(ilocal,:);
    TargetEdge(ilocal).edgeID = ilocal;
end

% Differentiating between the interior and exterior nodes
local_ext_nodes = nan(1, floor(length(TargetNode)));
local_int_nodes = nan(1, floor(length(TargetNode)));
global_int_nodes = nan(1, floor(length(TargetNode)));
ixn = 1;
iin = 1;
for itn = 1 : numel(target_nodes)
    local_tc = TargetNode(itn).connectivity;
    global_tc = TargetNode(itn).GlobalConnectivity;
    if local_tc ~= global_tc || global_tc == 1
        local_ext_nodes(ixn) = itn;
        ixn = ixn + 1;
    else
        local_int_nodes(iin) = itn;
        global_int_nodes(iin) = target_nodes(itn);
        iin = iin + 1;
    end
end
end

```

```

local_int_nodes(isnan(local_int_nodes)) = [];
global_int_nodes(isnan(global_int_nodes)) = [];
local_ext_nodes(isnan(local_ext_nodes)) = [];

% Differentiating between the interior and exterior edges
local_target_edges = 1: numel(target_edges);
local_ext_edges = [TargetNode(local_ext_nodes).linkID]';
local_int_edges = local_target_edges(~ismember(local_target_edges,
...
    local_ext_edges));
global_int_edges = target_edges(local_int_edges);

% truncating the target edges and nodes even further
TargetEdgeInt = TargetEdge(local_int_edges);
TargetNodeInt = TargetNode(local_int_nodes);

% removing exterior nodes from the interior node structure
for inr = 1 : numel(local_int_nodes)
    linkID = TargetNodeInt(inr).linkID;
    io1 = ismember(linkID, local_ext_edges);
    TargetNodeInt(inr).linkID(io1) = [];
    [~, loc] = ismember(TargetNodeInt(inr).linkID, local_int_edges);
    TargetNodeInt(inr).linkID = loc;
    TargetNodeInt(inr).connectivity =
numel(TargetNodeInt(inr).linkID);
end

% removing the exterior edges from the interior edge structure
for ier = 1 : length(TargetEdgeInt)
    [~, loc] = ismember(TargetEdgeInt(ier).endID, local_int_nodes);
    TargetEdgeInt(ier).endID = loc;
end

and % Running the main clean-up loop for the noi, as well as the nodes
% edges immediately surrounding the noi.
[TargetEdgeInt, TargetNodeInt] = MainLoop(TargetEdgeInt,
TargetNodeInt, 'Merge');

%%%%%%%%%%%%%%

% Locating the deleted edges/nodes
old_EdgeIDs = unique([TargetEdgeInt.edgeID]);
local_emissing = local_int_edges(~ismember(local_int_edges,
old_EdgeIDs));
local_epresent = local_int_edges(ismember(local_int_edges,
old_EdgeIDs));

global_emissing = target_edges(local_emissing);
global_epresent = target_edges(local_epresent);

old_NodeIDs = unique([TargetNodeInt.nodeID]);
local_nmissing = local_int_nodes(~ismember(local_int_nodes,
old_NodeIDs));

global_nmissing = target_nodes(local_nmissing);

% Converting to global indices
global_int_edges(ismember(global_int_edges, global_emissing)) = [];
local_int_edges(ismember(local_int_edges, local_emissing)) = [];

for ile = 1 : length(TargetEdgeInt)
    endID_local = local_int_nodes(TargetEdgeInt(ile).endID)';
    [~, loc] = find(ismember(local_int_nodes, endID_local));
    endID_global = global_int_nodes(loc);
    TargetEdgeInt(ile).endID = endID_global';
end

for iln = 1 : length(TargetNodeInt)
    linkID_local = local_int_edges(TargetNodeInt(iln).linkID);
    [~, loc] = find(ismember(local_int_edges, linkID_local));
    linkID_global = global_int_edges(loc)';

```

```

        TargetNodeInt(iln).linkID = linkID_global;
        after_linkID_vec = TargetNodeInt(iln).linkID;
        before_linkID_vec = TargetNodeInt(iln).GlobalLinkID;
        new_linkID = unique([after_linkID_vec, before_linkID_vec]);
        new_linkID = new_linkID(~ismember(new_linkID, global_emissing));
        TargetNodeInt(iln).linkID = new_linkID;
        TargetNodeInt(iln).connectivity =
numel(TargetNodeInt(iln).linkID);
    end

    TargetEdgeInt = rmfield(TargetEdgeInt, {'GlobalEdgeID', 'edgeID'});
    TargetNodeInt = rmfield(TargetNodeInt, {'GlobalNodeID', 'nodeID',
...
    'GlobalLinkID', 'GlobalConnectivity'});
    %
    Edge(global_int_edges) = TargetEdgeInt;
    Node(global_int_nodes) = TargetNodeInt;

    % Ensuring that the repalced edges are correctly flipped with the
    % appropriate end nodes.
    for ige = 1 : numel(global_epresent)
        eoi = global_epresent(ige);
        noi = Edge(eoi).endID;
        e1p = [Edge(eoi).xdata; ...
            Edge(eoi).ydata; ...
            Edge(eoi).zdata];
        e2p = [Node(noi).xdata; ...
            Node(noi).ydata; ...
            Node(noi).zdata];

        end1p = [e1p(:,1), e1p(:,end)];
        end2p = [e2p(:,1), e2p(:,end)];

        end1p_flip = flip1r(end1p);

        u1 = end1p_flip(:,1);
        u2 = end1p_flip(:,2);
        v1 = end2p(:,1);
        v2 = end2p(:,2);

        e = 0.2;

        if sum(u1<=v1+e & u1>=v1-e) == 3 && ...
            sum(u2<=v2+e & u2>=v2-e) == 3
            e1p = flip1r(e1p);
            Edge(eoi).xdata = e1p(1,:);
            Edge(eoi).ydata = e1p(2,:);
            Edge(eoi).zdata = e1p(3,:);
        end
    end

    [Edge, Node] = RemoveDeleted(Edge, Node, global_emissing,
global_nmissing, cleaner_type);

    end

end

Connectivity = [Node.connectivity];
nc = hist(Connectivity, 1:10);

fprintf(' %4i|%5i|%5i|%5i|%5i|%5i|%5i|%5i|%5i|%5i|\n', it, ...
    nc(1), nc(2), nc(3), nc(4), nc(5), nc(6), nc(7), nc(8), nc(9),
nc(10), length(Node));

shortID = find([Edge.length] < 1t, 1);
GoShort = ~isempty(shortID);

end

end

```

Appendix B: Supplementary information for electrical conductivity quantification

B.1 Benchmark for bulk electrical conductivity computation

Our finite-difference electrical conductivity calculator (FDECC) finds an approximate solution to the current continuity (Laplace) equation, given local conductivities each material. For a set of voxels connected in series, the approximation is perfectly accurate, but for a curved surface there is discretization error. We assess that error by computing the bulk electrical conductivity (σ_{bulk}) of a 15-pixel radius conductive sphere ($\sigma_1 = 0.06$ S/m) embedded in a relatively insulative $100 \times 100 \times 100$ voxel³ cube ($\sigma_2 = 10^{-5}$ S/m). The analytical solution for the bulk electrical conductivity follows the Maxwell-Garnett relation (Markov, 1999, Hughes, 2000), which is

$$\frac{\sigma_{\text{bulk}} - \sigma_2}{\sigma_{\text{bulk}} + 2\sigma_2} = \phi \frac{\sigma_1 - \sigma_2}{\sigma_1 + 2\sigma_2} \quad \text{B.1}$$

where ϕ is the phase fraction of the sphere, which is 0.0141 for the sphere and cube dimensions listed above. This is the same benchmark computation used by Zhan (2010) to validate their model. According Eqn. B.1, the analytical solution for σ_{bulk} is 0.058740 S/m, and σ_{bulk} from FDECC is 0.058709 S/m. The small error (0.05%) between the analytical and numerical solution suggests that FDECC accurately estimates the bulk electrical conductivity of the input olivine-melt and olivine-opx-melt geometries.

```

% ----- %
%                               wrapper script for FDECC
% ----- %

drName = [pwd, '/test_images/'];
fNameList = { ...
    'crossn.tif'
    'sphereInABox.tif'
};

sigmaList = [ ...
    7.53 0.009
];

% addendum = '_cropped(400)';
addendum = '';

nFile = length(fNameList);
for iSigma = 1 : size(sigmaList, 1)
    sigma = sigmaList(iSigma,:);
    sigmaStr = sprintf('%.3f-', sigma(:));
    sigmaStr = sigmaStr(1:end-1);
    sigmaBulk = zeros(size(fNameList, 1), 1);
    for iFile = 1 : nFile
        nRand = randsample(1:1e4, 1);
        sRoot = '/Users/kevinmiller/data/dc/results/temp/';
        sDir = sprintf( ...
            '%s%s/', sRoot, fNameList{iFile}(1:end-4), addendum);
        if exist(sDir, 'dir') == 0
            mkdir(sDir);
        end

        flowDir = 'Z';
        diaryName = sprintf( ...
            '%s%s_flow%s_sigma%s_%04i.out', ...
            sDir, ...
            fNameList{iFile}(1:end-4), ...
            flowDir, ...
            sigmaStr, ...
            nRand);

        if exist(diaryName, 'file') > 0
            delete(diaryName);
        end

        diary(diaryName);

        [~, nameComp] = system('hostname');
        fprintf('\n%s', nameComp);

        fprintf('\nLoading %s%s\n', drName, fNameList{iFile});

        G = uint8( ...
            Tif3DReader( ...
                drName, ...
                fNameList{iFile} ...
            ) ...
        );

        %
        % geomLim = 400;
        % domainLim = size(G);
        % domainOver = domainLim - geomLim;
        % if domainOver(1) > 0
        %     halfOver = floor(domainOver(1) / 2);
        %     G = G(halfOver:end-halfOver-1,:,:);
        % end
        % if domainOver(2) > 0
        %     halfOver = floor(domainOver(2) / 2);
        %     G = G(:,halfOver:end-halfOver-1,:);
        % end
        % if domainOver(3) > 0
        %     halfOver = floor(domainOver(3) / 2);

```

```

%         G = G(:, :, halfOver:end-halfOver-1);
%     end

%     if domainLim > geomLim
%         sizeOver(1) =
%         G(G == 0) = 2;
%         G = G(151:350, 151:350, 151:350);
%         G = G(51:450, 51:450, 51:450);

%         if numel(sigma) > 1
%             Model = dc3dn(G, flowDir, sigma);
%         else
%             Model = dc3d(G, flowDir, sigma);
%         end

%         saveresult(sDir, Model, nRand);
%         fprintf('\n');

%         diary off;

%         sigmaBulk(iFile) = Model.result.sigmaEff;
%     end

%     if numel(sigmaBulk) > 1
%         save( ...
%             sprintf( ...
%                 'sigma%s_%04i_sigmaBulk.mat', ...
%                 sigmaStr, ...
%                 nRand ...
%             ), ...
%             'sigmaBulk' ...
%         );
%     end
end

% ----- %

function IF = Tif3DReader(Dir, FileTif, varargin)

if ~isempty(varargin)
    if strcmp(varargin{1}, 'Plot')
%         cmd = varargin{1};
        islice = varargin{2};
        if ischar(islice) && strcmp(varargin{2}, 'All')
            else
                islice = varargin{2};
            end
        end
    end
end

% FileTif='rec_scoba_12_200x200x200_sample8_pc-melt_final.tif';
InfoImage=imfinfo([Dir, FileTif]);
mImage=InfoImage(1).Width;
nImage=InfoImage(1).Height;
NumberImages=length(InfoImage);
FinalImage=zeros(nImage,mImage,NumberImages,'uint16');

TifLink = Tiff([Dir, FileTif], 'r');
for i=1:NumberImages
    TifLink.setDirectory(i);
    FinalImage(:, :, i)=TifLink.read();
end
TifLink.close();
% FinalImage = double(FinalImage);

% getting the dimensions of the sample
% xloc = strfind(FileTif, 'x');
% xDim = str2num(FileTif(xloc(1)-3:xloc(1)-1));
% yDim = str2num(FileTif(xloc(2)-3:xloc(2)-1));
% zDim = str2num(FileTif(xloc(2)+1:xloc(2)+3));

```

```

IF = FinalImage;

% % Imported this section from online code
% % http://people.ece.cornell.edu/land/PROJECTS/Reconstruction/index.html
% %patch smoothing factor
% rfactor = 0.125;
% %isosurface size adjustment
% level = .8;
% %useful string constants
% c2 = 'facecolor';
% c1 = 'edgecolor';
%
% p=patch(isosurface(smooth3(FinalImage==1),level));
% reducepatch(p,rfactor)
% set(p,c2,[1,0,0],c1,'none');
%
% p=patch(isosurface(smooth3(FinalImage==2),level));
% reducepatch(p,rfactor)
% set(p,c2,[0,1,0],c1,'none');
% % spy(FinalImage(:,:,islice));
% [Xi, Yi, Zi] = meshgrid(0:1:xDim-1, 0:1:yDim-1, 0:1:zDim-1);
%
% % Xi = uint8(Xi);
% % Yi = uint8(Yi);
% % Zi = uint8(Zi);
% % fidbl = double(FinalImage);
% % figure(1); clf;
% % ImageData2D = FinalImage(:,:,islice);
% % fv = isosurface(fidbl, Xi, Yi, Zi);
% % slice(FinalImage, Xi, Yi, Zi);
% % colormap(jet);
% % bwi = im2bw(FinalImage(:,:,islice));
% % image(bwi);
% if ~isempty(varargin)
%     image(FinalImage(:,:,islice));
% end
% colormap(jet);

end

% ----- %

function Model = dc3dn(G, flowAxis, sigma, varargin)
%DC3DN 3-D direct current experiment simulation (for N conductivities).
%
% [MODEL] = dc3dn(x) conducts a direct current experiment on a label
% geometry. The finite difference method is used to solve the discrete
% Laplace equation within a specified binary image. A different electric
% potential is imposed at the inlet and outlet faces of the geometry, and
% a no flux condition is applied to the boundary box faces that are
% perpendicular to the direction of current. dc3dn can handle an
% arbitrary number of materials that have different conductivities.
% Either a direct or iterative approach is taken to solve the system of
% equations. The current density is then calculated using a centered
% difference gradient, volume-averaged, and then the effective electrical
% conductivity of the volume is calculated.
%
% [MODEL] = dc3dn(G, FLOWAXIS, SIGMA) conducts a direct current
% experiment on the 3-D label image G in the direction specified by the
% string FLOWAXIS. Conductivities are given by the vector SIGMA and are
% applied to materials specified by its index. Results are outputted to
% structure MODEL.
%
% Examples:
%     drName = pwd;
%     fName = 'crossn.tif';
%     G = logical(Tif3DReader(drName, fName));
%     Model = dc3dn(G, 'x', [1 .01]);
%
% Class support for input G:
%     uint8, uint16, single, double

```



```

% $Author: Kevin J. Miller $ $Date: 04-Feb-2015 09:25:40 $ $Revision: 1.0 $
% Copyright: Kevin J. Miller 2015

fprintf('\n-----');
fprintf('\n          Initiating Direct Current Experiment          \n');
fprintf('-----\n');

tic;
% initiating timer

% ----- Setting parameters ----- %

fprintf('\n%s\n', datestr(now));
% time-stamps the simulation
fprintf('\nFlow Direction: %s\n', flowAxis);
% printing the flow direction

Model = struct( ...
% allocating memory for structure
    'params', [], ...
    'geom',   [], ...
    'bids',   [], ...
    'lids',   []);

Model.params = loadparams(Model.params);
% loading parameters from text file

Model.params.flowAxis = flowAxis;
uMat = unique(G); uMat(uMat == 0) = [];
% checking that the number of conductances matches the number of materials
nMat = numel(uMat);
Model.params.nMat = nMat;
if nMat ~= numel(sigma)
    error('Number of conductances does not match the number of materials');
end
Model.params.flowAxis = flowAxis;
Model.params.sigma = sigma;

fprintf('\tConductivities:\t\t(');
fprintf(' %.3e', sigma(:)); fprintf(' ) [S/m]');

% ----- Loading the geometry ----- %

switch Model.params.flowAxis
% rotating geometry into position
    case 'Y'
        G = uint8(rotategeom_Gen2(G, Model.params.flowAxis, 1));
    case 'Z'
        G = uint8(rotategeom_Gen2(G, Model.params.flowAxis, 1));
end

Model.geom.G = G;

fprintf(...
    '\n\tDimensions:\t\t\tix%i\n', ...
    size(G, 1), size(G, 2), size(G, 3));

Model = impreprocessn(Model, 'Enclose', 'Refine', Model.params.cres);
% preprocessing image

Model.geom.dim = size(Model.geom.G);
% geometry dimensions (in pixels)
Model.geom.bounds = [
% boundaries of geometry
    1 Model.geom.dim(1), ...
    1 Model.geom.dim(2), ...
    1 Model.geom.dim(3) ...
];
Model.geom.L = Model.geom.dim(1);
% length of geometry
Model.params.ndof = prod( size(Model.geom.G) - 2 );
% number of degrees of freedom

```

```

% ----- Begin main block ----- %

Model = discretizen(Model);
% discretizing geometry

[connect, connectBound, connectSigma, Model] = assembleconnectn(Model);
% assembling connectivity matrix

A = assemblematrixn(Model, connect, connectBound, connectSigma);
% building matrix with boundary conditions

Model.lids.inlet = [];
% clearing unnecessary variables from structure
Model.lids.outlet = [];
Model.lids.noFlux = [];

b = assembleloadvectorn(Model, connect, connectBound);
% building load vector with boundary conditions

clear connect connectBound connectSigma;

sider = size(Model.geom.G, 1) - 2;
x = 1 : sider;
vExp = (-1 * (Model.params.V_inlet - Model.params.V_outlet) / sider) * ...
    x + Model.params.V_inlet;
x0 = repmat( ...
    vExp', ...
    [1 size(Model.geom.G, 2) - 2 size(Model.geom.G, 3) - 2]);
x0 = x0(:);

solveStruct = struct( ...
    'droptol', Model.params.droptol, ...
    'thresh', Model.params.thresh, ...
    'udiag', Model.params.udiag, ...
    'soltype', Model.params.soltype, ...
    'maxiter', Model.params.maxiter, ...
    'reltol', Model.params.reltol, ...
    'x0', x0 ...
);

clear x0 x0;

dumpPath = [pwd, '/dump/'];
% dumping structure to hard disk
if ~exist(dumpPath, 'dir')
    mkdir(dumpPath)
end
save([dumpPath, 'Model.mat'], 'Model', '-v7.3');
clear Model;

x = dcsolvern(A, b, solveStruct);
% solving linear system

load([dumpPath, 'Model.mat']);
% recovering structure from hard disk
delete([dumpPath, 'Model.mat']);

% ----- End main block ----- %

lidInteriorAll = [];
for iMat = 1 : nMat
    lidInteriorAll = [lidInteriorAll; Model.lids.interior{iMat}];
end
lidInteriorAll = sort(lidInteriorAll);

v = zeros(prod(Model.geom.dim), 1);
v(lidInteriorAll) = x;

for iMat = 1 : nMat
    v(Model.bids.inlet{iMat}) = Model.params.V_inlet;
    v(Model.bids.outlet{iMat}) = Model.params.V_outlet;
end

```

```

end

V = reshape(v, Model.geom.dim(1), Model.geom.dim(2), Model.geom.dim(3));
Model.result.V = V;

clear G A x b lidInteriorAll v V;

Model = postprocessingn(Model);
% conducting the postprocessing

fprintf('\n');
toc;

fprintf('\n-----');
fprintf('\n                        End of Simulation                \n');
fprintf('-----\n');

end

% ----- %

function A = loadparams(A)

fprintf('\nSetting parameters for model...\n');

% Parameters that modify the input geometry
A.rmispurs = 0;
A.enclose = 1;
A.cres = 1;
A.rmislnds = 0;
A.islthresh = 100;
A.addinout = 1;

% Parameters that modify boundary conditions and material properties
A.V_inlet = 2;
A.V_outlet = 1;
A.V_vn = 0;
A.h = 1;
% A.sigma = [10 .009];
% A.nMat = numel(A.sigma);

% Parameters that modify preconditioner options
A.droptol = 1e-3;
A.thresh = 0;
A.udiag = 1;

% Parameters that modify solver options
A.soltype = 'iter';
A.iterkeep = 2;
A.maxiter = 1e4;
A.reltol = 1e-7;

fprintf(sprintf('\n\tRemove Spurs:\t\t\t\t\t%i', A.rmispurs));
fprintf(sprintf('\n\tEnclose geometry:\t\t\t\t\t%i', A.enclose));
fprintf(sprintf('\n\tRemove Islands (<i):\t\t\t\t\t%i', A.islthresh,
A.rmislnds));
fprintf(sprintf('\n\tAppend inlet/outlet:\t\t\t\t\t%i', A.addinout));
fprintf(sprintf('\n\tResample geometry:\t\t\t\t\t%i\n', A.cres));
fprintf(sprintf('\n\tInlet potential:\t\t\t\t\t%i [V]', A.V_inlet));
fprintf(sprintf('\n\tOutlet potential:\t\t\t\t\t%i [V]', A.V_outlet));
fprintf(sprintf('\n\tSpacing:\t\t\t\t\t%i [m]', A.h));
% fprintf(sprintf('\n\tConductivities:\t\t\t\t\t%.0e, %.0e [S/m]\n', A.sigma(1),
A.sigma(2)));
fprintf(sprintf('\n\tMaximum iterations:\t\t\t\t\t%i', A.maxiter));
fprintf(sprintf('\n\tRelative tolerance:\t\t\t\t\t%.0e\n', A.reltol));

% ----- %

```

```

function G_rot = rotategeom_Gen2(G, flipAxis, flipDir)
switch flipAxis
    case 'Y'
        switch flipDir
            case 1
                G_rot = flipdim( permute(G, [2 1 3]), 2 );
            case -1
                G_rot = permute( flipdim(G, 2), [2 1 3] );
        end
    case 'Z'
        switch flipDir
            case 1
                G_rot = flipdim( permute(G, [3 2 1]), 3 );
            case -1
                G_rot = permute( flipdim(G, 3), [3 2 1] );
        end
    end
end

% ----- %

function Model = improcessn(Model, varargin)
% ----- %
% This subroutine preprocesses 2D binary image data for running CFD
% simulations.

% if ~isempty(varargin)
%     GoRmSpurs = ~isempty(find(strcmp(varargin, 'Remove Spurs'), 1));
%     GoRefine = ~isempty(find(strcmp(varargin, 'Refine'), 1));
%     GoEnclose = ~isempty(find(strcmp(varargin, 'Enclose'), 1));
%     GoRmIslands = ~isempty(find(strcmp(varargin, 'Remove Islands'), 1));
%     GoOpenInlets = ~isempty(find(strcmp(varargin, 'Open Inlets'), 1));
%     if GoRmSpurs
%         rms_loc = find(strcmp('Remove Spurs', varargin), 1);
%         tconn = varargin{rms_loc+1};
%     end
%     if GoRefine
%         ref_loc = find(strcmp('Refine', varargin), 1);
%         trefine = varargin{ref_loc+1};
%     end
%     if GoEnclose
%         flowInd = find(Model.params.flowVec, 1);
%     end
%     if GoRmIslands
%         islandth_loc = find(strcmp('Remove Islands', varargin), 1);
%         islandth = varargin{islandth_loc+1};
%     end
% else
%     GoRmSpurs = 0;
%     GoRefine = 0;
%     GoEnclose = 0;
%     GoRmIslands = 0;
%     GoOpenInlets = 0;
% end

% if Model.params.
%     Model.geom.G = OpenInlets(Model.geom.G);
% end
% if GoRmSpurs
%     Model.geom.G = RmSpurs(Model.geom.G, tconn);
% end
Model.geom.G0 = Model.geom.G;
if Model.params.cres > 1
    Model.geom.G = imresampen(Model.geom.G, Model.params.cres);
    Model.geom.G0 = Model.geom.G;
end
if Model.params.addinout
    Model.geom.G = addinout(Model.geom.G);
end
% if Model.params.enclose

```

```

% %      Model.geom.G0 = Model.geom.G;
% Model.geom.G = dc_ImEnclose3D(Model.geom.G);
% else
% Model.geom.G0 = Model.geom.G;
% end
if Model.params.rmislands
%      conn = 8;
%      Model.geom.G = rmislands(Model.geom.G,
3*Model.params.cres*Model.params.islthresh);
end

end

% ----- %

function G = addinout(G)

G = padarray(G, [1 1 1], 'replicate');

% inletCopy = G(1,:,:);
% outletCopy = G(end,:,:);
%
% G = cat(1, inletCopy, G);
% G = cat(1, G, outletCopy);

end

% ----- %

function newG = rmislands(G, thresh)

fprintf('\tRemoving islands(<%i)...', thresh);

CC = bwconncomp(G, 6);
F = zeros(size(G), 'uint16');
ival = uint16(1);
nIsl = numel(CC.PixelIdxList);
for iIsl = 1 : nIsl
    cIsl = CC.PixelIdxList{iIsl};
    if size(cIsl, 1) > thresh
        F(CC.PixelIdxList{iIsl}) = ival;
        ival = ival + 1;
    end
end

newG = F > 0;

nisl = sum(G(:)) - sum(newG(:));

fprintf('%i pixels modified', nisl);

end

% ----- %

function Model = discretizen(Model)

% Discretizes image that consists of an arbitrary number of materials.

fprintf('\nDiscretizing geometry...');

Model.bids = struct( ...
% storing binary images and linear ID's in 'Model' structure
    'inlet',    [], ...
    'outlet',   [], ...
    'solLiq',   [], ...
    'noFlux',   [], ...
    'interior', [], ...
    'inside',   []);

Model.lids = struct( ...
% storing binary images and linear ID's in 'Model' structure

```

```

        'inlet',    [], ...
        'outlet',   [], ...
        'solLiq',   [], ...
        'noFlux',   [], ...
        'interior', []];

nMat = Model.params.nMat;

for iMat = 1 : nMat

    noFlux = Model.geom.G == iMat;
    noFlux(:,2:Model.geom.dim(2)-1,2:Model.geom.dim(3)-1) = 0;

    inside = Model.geom.G == iMat;
    % binary image of inside nodes

    inlet = false( ...
    % allocating memory for binary image of inlet nodes
        Model.geom.bounds(2), ...
        Model.geom.bounds(4), ...
        Model.geom.bounds(6));

    outlet = false( ...
    % allocating memory for binary image of outlet nodes
        Model.geom.bounds(2), ...
        Model.geom.bounds(4), ...
        Model.geom.bounds(6));

    inlet(1,:,:)= inside(1,:,:);
    % binary image of inlet nodes
    outlet(Model.geom.bounds(2),:,:) = inside(Model.geom.bounds(2),:,:);
    % binary image of outlet nodes

    inside_test =         inside;
    inside_test(:,1,:) = 0;
    inside_test(:,end,:) = 0;
    inside_test(:,1) = 0;
    inside_test(:,end) = 0;

    solLiqStrel(:,1) = [0 0 0; 0 1 0; 0 0 0];
    solLiqStrel(:,2) = [0 1 0; 1 1 1; 0 1 0];
    solLiqStrel(:,3) = [0 0 0; 0 1 0; 0 0 0];

    solLiq = imdilate(inside_test, solLiqStrel) & ~inside_test;
    % binary image of solid-liquid boundary boundary nodes

    inlet =      inlet & ~noFlux;
    outlet =     outlet & ~noFlux;

    solLiq(1,:,:)= 0;
    % removing inlet positions from solLiq
    solLiq(Model.geom.bounds(2),:,:) = 0;
    % removing outlet positions from solLiq
    solLiq(:,1,:)= 0;
    solLiq(:,Model.geom.bounds(4),:)= 0;
    solLiq(:,1) = 0;
    solLiq(:,Model.geom.bounds(6)) = 0;

    interior = inside & ~inlet & ~outlet & ~noFlux;
    % removing solLiq, inlet, and outlet nodes from 'inside' binary image

    Model.bids = storestructn(Model.bids, { ...
    % storing binary images and linear ID's in 'Model' structure
        'inlet'      inlet
        'outlet'     outlet
        'solLiq'     solLiq
        'noFlux'     noFlux
        'interior'   interior
        'inside'     inside
    });

```

```

        Model.lids = storestructn(Model.lids, { ...
% storing binary images and linear ID's in 'Model' structure
        'inlet'      find(inlet)
        'outlet'     find(outlet)
        'solLiq'     find(solLiq)
        'noFlux'     find(noFlux)
        'interior'   find(interior)
        });

end

end

% ----- %

function S = storestructn(S, storeName, varargin)

% if ~isempty(varargin)
%     if strcmpi(varargin, 'append')
%         appendSwitch = 1;
%     else
%         appendSwitch = 0;
%     end
% else
%     appendSwitch = 0;
% end

sizeStruct = structfun(@(x) size(x, 2), S);
uMat = unique(sizeStruct);
if numel(uMat) > 1
    error('Sizes of structure fields are not consistent');
end
cMat = uMat + 1;
nstr = length(storeName);
for istr = 1 : nstr
    S.(storeName{istr,1}){cMat} = storeName{istr,end};
end

end

% ----- %

function [connect, connectBound, connectSigma, Model] = ...
    assembleconnectn(Model)

% Subroutine for assembling the connectivity matrix.

fprintf('\n\nAssembling connectivity matrix...');

nMat = Model.params.nMat;

connect = cell(nMat, 1);
connectSigma = cell(1, Model.params.nMat);

sigmaAll = zeros(size(Model.geom.G));
for iMat = 1 : Model.params.nMat
    sigmaAll(Model.bids.inside{iMat}) = Model.params.sigma(iMat);
    sigmaAll(Model.bids.noFlux{iMat}) = Model.params.sigma(iMat);
end

sigmaAll(Model.bids.noFlux{1} | Model.bids.noFlux{2}) = 0;

Model.sigmaAll = sigmaAll;

connectBound = struct( ...
    'isInlet',      [], ...
    'isOutlet',     [], ...
    'isSolLiq',     [], ...
    'isNoFlux',     [], ...
    'isConnBound',  []);

for iMat = 1 : nMat

```

```

[iInBox, jInBox, kInBox] = ind2sub(...
    size(Model.bids.interior{iMat}), ...
    find(Model.bids.interior{iMat} == 1));
% coordinates of the center nodes
ijkInBox = [iInBox, jInBox, kInBox];

xMinus = [ijkInBox(:,1) - 1, ijkInBox(:,2),      ijkInBox(:,3)    ];
% coordinates of the west-shifted nodes
xPlus = [ijkInBox(:,1) + 1, ijkInBox(:,2),      ijkInBox(:,3)    ];
% coordinates of the east-shifted nodes
yMinus = [ijkInBox(:,1),      ijkInBox(:,2) - 1, ijkInBox(:,3)    ];
% coordinates of the south-shifted nodes
yPlus = [ijkInBox(:,1),      ijkInBox(:,2) + 1, ijkInBox(:,3)    ];
% coordinates of the north-shifted nodes
zMinus = [ijkInBox(:,1),      ijkInBox(:,2),      ijkInBox(:,3) - 1];
% coordinates of the south-shifted nodes
zPlus = [ijkInBox(:,1),      ijkInBox(:,2),      ijkInBox(:,3) + 1];
% coordinates of the north-shifted nodes

ctrLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    ijkInBox(:,1), ...
    ijkInBox(:,2), ...
    ijkInBox(:,3));
% linear indices of the center nodes

xMinusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    xMinus(:,1), ...
    xMinus(:,2), ...
    xMinus(:,3));
west-shifted nodes                                     % linear indices of the

xPlusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    xPlus(:,1), ...
    xPlus(:,2), ...
    xPlus(:,3));
east-shifted nodes                                     % linear indices of the

yMinusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    yMinus(:,1), ...
    yMinus(:,2), ...
    yMinus(:,3));
south-shifted nodes                                    % linear indices of the

yPlusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    yPlus(:,1), ...
    yPlus(:,2), ...
    yPlus(:,3));
north-shifted nodes                                    % linear indices of the

zMinusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    zMinus(:,1), ...
    zMinus(:,2), ...
    zMinus(:,3));
south-shifted nodes                                    % linear indices of the

zPlusLids = sub2ind( ...
    size(Model.bids.interior{iMat}), ...
    zPlus(:,1), ...
    zPlus(:,2), ...
    zPlus(:,3));
north-shifted nodes                                    % linear indices of the

connect{iMat} = [ ...
    ctrLids, ...
    xMinusLids, ...
    xPlusLids, ...
    yMinusLids, ...
    yPlusLids, ...
    zMinusLids, ...

```



```

        zPlusLids];
% connectivity matrix of interior nodes

    connectIsInlet = ismembc( ...
        connect{iMat}, ...
        Model.lids.inlet{iMat} ); % logical array showing
connectivities that are located on inlet
    connectIsOutlet = ismembc(...
        connect{iMat}, ...
        Model.lids.outlet{iMat}); % logical array showing
connectivities that are located on outlet
    connectIsSolLiq = ismembc( ...
        connect{iMat}, ...
        Model.lids.solLiq{iMat}); % logical array showing
connectivities that are located on solid-pore interface
    connectIsNoFlux = ismembc( ...
        connect{iMat}, ...
        Model.lids.noFlux{iMat}); % logical array showing
connectivities that are located on solid-pore interface

    connectIsInlet(connectIsInlet(:,1),:) = 0;
% removing nodes that are part of 'inlet' from connectivity matrix
    connectIsOutlet(connectIsOutlet(:,1),:) = 0;
% removing nodes that are part of 'otlet' from connectivity matrix
    connectIsNoFlux(connectIsNoFlux(:,1),:) = 0;
% removing nodes that are part of 'otlet' from connectivity matrix

    isConnBound = ( ...
% logical index of nodes that are connected to boundary nodes
        sum(connectIsInlet, 2) ...
        + sum(connectIsOutlet, 2) ...
        + sum(connectIsNoFlux, 2)) > 0;

    connectSigma{iMat} = sigmaAll(connect{iMat});

    connectBound = storestructn(connectBound, { ...
        'isInlet'      connectIsInlet
        'isOutlet'     connectIsOutlet
        'isSolLiq'     connectIsSolLiq
        'isNoFlux'     connectIsNoFlux
        'isConnBound'  isConnBound
    });

end

Model.sigmaAll = single(Model.sigmaAll);

end

% ----- %

function A = assemblematrixn(Model, connect, connectBound, connectSigma)
% Subroutine for assembling the coefficient matrix.
fprintf('\n\nAssembling coefficient matrix...');
sigmaExp = @(s1, s2) (2 * s1 .* s2) ./ (s1 + s2);
nStencilPts = 7;
% size of finite-difference stencil

maxMatBounds = repmat(prod(Model.geom.dim), [1 2]);
% maximum matrix bounds for stiffness matrix
nonZeroMax = nStencilPts*maxMatBounds(1);
% mamimum possible number of non-zero element in stiffness matrix

A = spalloc(maxMatBounds(1), maxMatBounds(2), nonZeroMax);
augList = [];

for iMat = 1 : Model.params.nMat

```

```

sigmaNoFlux = connectSigma{iMat}.*(connectBound.isNoFlux{iMat});
sumSigmaNoFlux = sum(sigmaNoFlux, 2);
notC_isNoFlux = ~connectBound.isNoFlux{iMat};

notC_isInlet = ~connectBound.isInlet{iMat};
notC_isOutlet = ~connectBound.isOutlet{iMat};

cCtr = -1 * ( ...
    sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,2)) ...
    + sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,3)) ...
    + sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,4)) ...
    + sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,5)) ...
    + sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,6)) ...
    + sigmaExp(connectSigma{iMat}(:,1), connectSigma{iMat}(:,7)) ...
    + sumSigmaNoFlux;

cWest = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1) .* ...
    notC_isInlet(:,1) .* notC_isOutlet(:,1), ...
    connectSigma{iMat}(:,2) .* notC_isNoFlux(:,2) .* ...
    notC_isInlet(:,2) .* notC_isOutlet(:,2) ...
);
cEast = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1) .* ...
    notC_isInlet(:,1) .* notC_isOutlet(:,1), ...
    connectSigma{iMat}(:,3) .* notC_isNoFlux(:,3) .* ...
    notC_isInlet(:,3) .* notC_isOutlet(:,3) ...
);
cSouth = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1), ...
    connectSigma{iMat}(:,4) .* notC_isNoFlux(:,4) ...
);
cNorth = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1), ...
    connectSigma{iMat}(:,5) .* notC_isNoFlux(:,5) ...
);
cLower = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1), ...
    connectSigma{iMat}(:,6) .* notC_isNoFlux(:,6) ...
);
cUpper = sigmaExp( ...
    connectSigma{iMat}(:,1) .* notC_isNoFlux(:,1), ...
    connectSigma{iMat}(:,7) .* notC_isNoFlux(:,7) ...
);

A = A + sparse(repmat(connect{iMat}(:,1), [7 1]), ...
    [connect{iMat}(:,1); ...
    connect{iMat}(:,2); ...
    connect{iMat}(:,3); ...
    connect{iMat}(:,4); ...
    connect{iMat}(:,5); ...
    connect{iMat}(:,6); ...
    connect{iMat}(:,7)], ...
    [cCtr; cWest; cEast; cSouth; cNorth; cLower; cUpper], ...
    maxMatBounds(1), maxMatBounds(2), nonZeroMax);

augList = cat(1, augList, ...
    [Model.lids.inlet{iMat}; ...
    Model.lids.outlet{iMat}; ...
    Model.lids.noFlux{iMat}]);

end

allList = 1:prod(Model.geom.dim);
augBidList = ismember(allList, augList);
intList = allList(~augBidList);

A = A(:,intList);
A = A.';
A = A(:,intList);
A = A.';

```

```

ndof = size(A, 1);
fprintf('\n\n\tNumber of degrees of freedom: %i\n', ndof);
end

% ----- %

function b = assembleloadvectorn(Model, connect, connectBound)
% Subroutine for assembling the load vector

npts = prod(Model.geom.dim);
bkeep = [];

b = spalloc(npts, 1, Model.params.ndof);
for iMat = 1 : Model.params.nMat;
    bvals = -1 * Model.params.sigma(iMat) * ( ...
% applying Dirichlet and Neumann boundary conditions
        Model.params.V_inlet * sum(connectBound.isInlet{iMat}, 2) + ...
        Model.params.V_outlet * sum(connectBound.isOutlet{iMat}, 2));

    b = b + sparse( ...
% forming sparse load vector
        connect{iMat}(:,1), ...
        ones(size(bvals)), ...
        bvals, ...
        npts, ...
        1, ...
        size(bvals, 1));

    bkeep = [bkeep; Model.lids.interior{iMat}];
end

bkeep = sort(bkeep);
b = b(bkeep);
% removing pixels that belong to the boundary or solid material
Model.lids.interior = [];
end

% ----- %

function x = dcsolvern(A, b, solveStruct)
% Subroutine for setting up and initializing the preconditioner and solver.
switch solveStruct.soltype
    case 'direct'
        fprintf('\nDirect solver: Matlab %s', '"');
        x = A\b;
    case 'iter'
        fprintf('\nPreconditioning matrix...\n');
        fprintf('\n\tPreconditioner: Incomplete Choleski Factorization');

        iluStruct = struct(...
            'type', 'ict', ...
            'droptol', solveStruct.droptol, ...
            'shape', 'lower' ...
        );

        L = ichol(-1*A, iluStruct);

```

```

fprintf('\n\tdroptol: %.1e\n', solveStruct.droptol);
fprintf('\nInitiating solver...\n');
fprintf('\n\tIterative solver: PCG\n\treltol: %.1e\n', ...
    solveStruct.reltol);

[x, flag, rr1, iter, relNorm] = pcg(...
    -1*A, ...
    -1*b, ...
    solveStruct.reltol, ...
    solveStruct.maxiter, ...
    L, L', ...
    solveStruct.x0);

switch flag
case 0
    fprintf('\n\tPCG converged to the desired tolerance %.1e
within %i iterations.\n', solveStruct.reltol, numel(relNorm));
case 1
    error('\n\tPCG iterated %i times but did not converge.\n',
solveStruct.maxiter);
case 2
    error('\n\tPreconditioner was ill-conditioned.\n');
case 3
    error('\n\tPCG stagnated.\n');
end

figure(1); clf;
plot(1:numel(relNorm), relNorm, '-o');
set(gca, 'YScale', 'log');
title('Convergence');
xlabel('Iterations');
ylabel('Relative Norm');

end

end

% ----- %

function Model = postprocessingn(Model)

% Subroutine for postprocessing the scalar electric potential data to
% obtain the bulk electrical conductivity for the geometry.

fprintf('\nPost-processing...\n');

% Post-processing
E = cell(1, 3);
J = cell(1, 3);

V = Model.result.V(:, 2:end-1, 2:end-1);
Model.result.V = V;

G = Model.geom.G(:, 2:end-1, 2:end-1);

gradVXctr = V(2:end, :, :) - V(1:end-1, :, :);
gradVYctr = V(:, 2:end, :) - V(:, 1:end-1, :);
gradVZctr = V(:, :, 2:end) - V(:, :, 1:end-1);

ex = gradVXctr;
ey = gradVYctr;
ez = gradVZctr;

ex = -1 * ex;
ey = -1 * ey;
ez = -1 * ez;

sigmaExp = @(s1, s2) (2 * s1 .* s2) ./ (s1 + s2);
sigmaShiftX = sigmaExp(Model.sigmaAll(2:end, 2:end-1, 2:end-1), ...
    Model.sigmaAll(1:end-1, 2:end-1, 2:end-1));
sigmaShiftY = sigmaExp(Model.sigmaAll(:, 3:end-1, 2:end-1), ...
    Model.sigmaAll(:, 2:end-1, 2:end-1));

```

```

    Model.sigmaAll(:,2:end-2,2:end-1));
sigmaShiftZ = sigmaExp(Model.sigmaAll(:,2:end-1,3:end-1), ...
    Model.sigmaAll(:,2:end-1,2:end-2));

jx = ex .* sigmaShiftX;
jy = ey .* sigmaShiftY;
jz = ez .* sigmaShiftZ;

switch Model.params.flowAxis
    case 'X'

        rotG = Model.geom.G;
        rotG = rotG(:,2:end-1,2:end-1);

        E{1} = ex;
        E{2} = ey;
        E{3} = ez;

        J{1} = jx;
        J{2} = jy;
        J{3} = jz;

    case 'Y'

        rotG = flip(permute(Model.geom.G, [2 1 3]), 1);
        rotG = rotG(2:end-1,:,2:end-1);

        v = flip(permute(v, [2 1 3]), 1);

        E{1} = flip(permute(ey, [2 1 3]), 1);
        E{2} = flip(permute(ex, [2 1 3]), 1);
        E{3} = flip(permute(ez, [2 1 3]), 1);

        J{1} = flip(permute(jy, [2 1 3]), 1);
        J{2} = flip(permute(jx, [2 1 3]), 1);
        J{3} = flip(permute(jz, [2 1 3]), 1);

    case 'Z'

        rotG = flip(permute(Model.geom.G, [3 2 1]), 1);
        rotG = rotG(2:end-1,2:end-1,:);

        v = flip(permute(v, [3 2 1]), 1);

        E{1} = flip(permute(ez, [3 2 1]), 1);
        E{2} = flip(permute(ey, [3 2 1]), 1);
        E{3} = flip(permute(ex, [3 2 1]), 1);

        J{1} = flip(permute(jz, [3 2 1]), 1);
        J{2} = flip(permute(jy, [3 2 1]), 1);
        J{3} = flip(permute(jx, [3 2 1]), 1);

end

dVdL = -1 * (Model.params.V_outlet - Model.params.V_inlet) / ...
    (Model.geom.L - 1);

switch Model.params.flowAxis
    case 'X'
        jxAvg = (1 / prod(size(J{1}))) * sum(sum(sum(J{1})));
        sigmaEff = jxAvg / dVdL;
    case 'Y'
        jyAvg = (1 / prod(size(J{2}))) * sum(sum(sum(J{2})));
        sigmaEff = jyAvg / dVdL;
    case 'Z'
        jzAvg = (1 / prod(size(J{3}))) * sum(sum(sum(J{3})));
        sigmaEff = jzAvg / dVdL;
end

Model.result = storestruct(Model.result, { ...
    'sigmaEff' sigmaEff
    'V' v

```

```

        'E' E
        'J' J
        'rotG' rotG
    });

fprintf('\n\tBulk electrical conductivity: %.4e\n', ...
        Model.result.sigmaEff);

end

% ----- %

function Model = modifygeom(Model)

fprintf('\nModifying geometry...\n');

% fprintf('\n\tDirectory:\t%s', drName);
% fprintf('\n\tFile:\t\t%s', fName);
fprintf('\n\tDimensions:\t%ix%ix%i\n', size(Model.geom.G, 1),
        size(Model.geom.G, 2), size(Model.geom.G, 3));

switch Model.params.flowAxis
    case 'Y'
        Model.geom.G = uint8(rotategeom_Gen2(Model.geom.G,
        Model.params.flowAxis, 1));
    case 'Z'
        Model.geom.G = uint8(rotategeom_Gen2(Model.geom.G,
        Model.params.flowAxis, 1));
end

fprintf('\n\tPreprocessing image\n');

if Model.params.cres > 1
    Model.geom.G = imresample(Model.geom.G, Model.params.cres);
    fprintf('\n');
end
if Model.params.enclose
    Model.geom.G = imenclose(Model.geom.G);
    fprintf('\n');
end
if Model.params.rmislans
    Model.geom.G = rmislans(Model.geom.G,
    3*Model.params.cres*Model.params.islthresh);
    fprintf('\n');
end
if Model.params.rmspurs
    Model.geom.G = rmspurs(Model.geom.G);
    fprintf('\n');
end
if Model.params.rminletspurs
    Model.geom.G = rminletspurs(Model.geom.G);
    fprintf('\n');
end

nDim = numel(size(Model.geom.G));
switch nDim
    case 2
        Model.geom.dim = size(Model.geom.G);
        % geometry dimensions (in pixels)

        Model.geom.bounds = [ ...
        % boundaries of geometry
            1 Model.geom.dim(1), ...
            1 Model.geom.dim(2) ...
        ];
    case 3
        Model.geom.dim = size(Model.geom.G);
        % geometry dimensions (in pixels)

        Model.geom.bounds = [ ...
        % boundaries of geometry
            1 Model.geom.dim(1), ...

```

```

        1 Model.geom.dim(2), ...
        1 Model.geom.dim(3) ...
    ];
end
Model.geom.L = Model.geom.dim(1);
% length of geometry

end

% ----- %

function G3 = imresample(G, trefine)
fprintf('\n\tResampling image to ');
G1 = imresize(G, trefine, 'nearest');
G2 = logical(rotategeom_Gen2(G1, 'Z', 1));
G2 = imresize(G2, [trefine*size(G2, 1) size(G2, 2)], 'nearest');
G3 = logical(flip(permute(G2, [3 2 1]), 1));
fprintf('%ix%ix%i', size(G3, 1), size(G3, 2), size(G3, 3));
end

% ----- %

function newG = imenclose(G)
fprintf('\tEnclosing geometry...');
% newG = G;
nDim = numel(size(G));
switch nDim
    case 2
        newG = padarray(G, [0 1]);
    case 3
        newG = padarray(G, [0 1 1]);
end
% newG(:,1,:) = 0;
% newG(:,end,:) = 0;
% newG(:, :, 1) = 0;
% newG(:, :, end) = 0;
nMod = abs(sum(G(:)) - sum(newG(:)));
fprintf('%i pixels were modified', nMod);
end

% ----- %

function newG = rmislands(G, thresh)
fprintf('\tRemoving islands(<%i)...', thresh);
CC = bwconncomp(G, 6);
F = zeros(size(G), 'uint16');
ival = uint16(1);
nIsl = numel(CC.PixelIdxList);
for iIsl = 1 : nIsl
    cIsl = CC.PixelIdxList{iIsl};
    if size(cIsl, 1) > thresh
        F(CC.PixelIdxList{iIsl}) = ival;
        ival = ival + 1;
    end
end
newG = F > 0;

```

```

    nisl = sum(G(:)) - sum(newG(:));
    fprintf('%i pixels modified', nisl);
end

% ----- %

function G = rmispurs(G)
    fprintf('\tRemoving spurs...');

    SE(:,:,1) = [0 0 0; 0 1 0; 0 0 0];
    SE(:,:,2) = [0 1 0; 1 0 1; 0 1 0];
    SE(:,:,3) = [0 0 0; 0 1 0; 0 0 0];

    % SE1 = [1 0 1];
    % SE2 = [1; 0; 1];
    % SE3 = cat(3, 1, 0, 1);

    tempG = double(G);

    nSpurs = 0;
    GoOn = 1;

    while GoOn > 0
        C = convn(tempG, SE, 'same');
        C = C.*tempG;
        spurs = (C == 1);
        tempG(spurs) = 0;
        nSpurs_temp = sum(spurs(:));
        if nSpurs_temp == 0
            GoOn = 0;
        end
        nSpurs = nSpurs + nSpurs_temp;
    end

    G = logical(tempG);

    fprintf(sprintf('%i pixels removed.', nSpurs));

end

% ----- %

function newG = rminletspurs(G)

    notG = ~G;
    neighborInletSlice = notG(2,:,:);
    neighborOutletSlice = notG(end-1,:,:);
    notG(1,:) = neighborInletSlice;
    notG(end,:) = neighborOutletSlice;
    newG = ~notG;

end

% ----- %

function borderBW = findborder(BW, varargin)

    if ~isempty(varargin)
        if strcmpi(varargin{1}, 'outside') % | 'outside'
            BW = ~BW;
        end
    end

    conn = conndef(3,'minimal');
    erodeBW = imerode(BW, conn);
    borderBW = BW & ~erodeBW;

end

```



```

% ----- %

function S = storestruct(S, storeName, varargin)

nstr = size(storeName, 1);
for istr = 1 : nstr
    S.(storeName{istr,1}) = storeName{istr,end};
end

end

% ----- %

function Model = assembleconn(Model)

fprintf('\n\nAssembling connectivity matrix...');

[iInBox, jInBox, kInBox] = ind2sub(size(Model.geom.G), Model.lids.interior);
% coordinates of the center nodes

xMinus = [iInBox - 1, jInBox,      kInBox      ];
% coordinates of the west-shifted nodes
xPlus = [iInBox + 1, jInBox,      kInBox      ];
% coordinates of the east-shifted nodes
yMinus = [iInBox,      jInBox - 1, kInBox      ];
% coordinates of the south-shifted nodes
yPlus = [iInBox,      jInBox + 1, kInBox      ];
% coordinates of the north-shifted nodes
syMinus = [ijInBox(:,1), ijInBox(:,2)
- 1];
% coordinates of the south-shifted
nodes
zMinus = [iInBox,      jInBox,      kInBox - 1];
% coordinates of the south-shifted nodes
zPlus = [iInBox,      jInBox,      kInBox + 1];
%
coordinates of the north-shifted nodes

ctrLids = sub2ind(size(Model.geom.G), iInBox, jInBox, kInBox);
%
linear indices of the centers nodes

xMinusLids = sub2ind(size(Model.geom.G), xMinus(:,1), xMinus(:,2),
xMinus(:,3));
% linear indices of the west-shifted nodes
xPlusLids = sub2ind(size(Model.geom.G), xPlus(:,1), xPlus(:,2),
xPlus(:,3));
% linear indices of the east-shifted nodes
yMinusLids = sub2ind(size(Model.geom.G), yMinus(:,1), yMinus(:,2),
yMinus(:,3));
% linear indices of the south-shifted nodes
yPlusLids = sub2ind(size(Model.geom.G), yPlus(:,1), yPlus(:,2),
yPlus(:,3));
% linear indices of the north-shifted nodes
zMinusLids = sub2ind(size(Model.geom.G), zMinus(:,1), zMinus(:,2),
zMinus(:,3));
% linear indices of the south-shifted nodes
zPlusLids = sub2ind(size(Model.geom.G), zPlus(:,1), zPlus(:,2),
zPlus(:,3));
% linear indices of the north-shifted nodes

clear xMinus xPlus yMinus yPlus zMinus zPlus;

connect = [ctrLids, xMinusLids, xPlusLids, yMinusLids, yPlusLids,
zMinusLids, zPlusLids];
% connectivity matrix of interior
nodes

clear xMinusLids xPlusLids yMinusLids yPlusLids zMinusLids zPlusLids;

connectIsInlet = ismember(connect, Model.lids.inlet);
% logical array showing connectivities that are located on inlet
connectIsOutlet = ismember(connect, Model.lids.outlet);
% logical array showing connectivities that are located on outlet
connectIsSolLiq = ismember(connect, Model.lids.solLiq);
% logical array showing connectivities that are located on solid-pore
interface

connectIsInlet(connectIsInlet(:,1),:) = 0;
% removing nodes that are part of 'inlet' from connectivity matrix
connectIsOutlet(connectIsOutlet(:,1),:) = 0;
% removing nodes that are part of 'otlet' from connectivity matrix

```

```

connectIsSolLiq(connectIsSolLiq(:,1),:) = 0;
% removing nodes that are part of 'solLiq' from connectivity matrix

isConnBound = ( ...
% logical index of nodes that are connected to boundary nodes
    sum(connectIsInlet, 2) ...
    + sum(connectIsOutlet, 2) ...
    + sum(connectIsSolLiq, 2)) > 0;

connectBound = struct( ...
    'isInlet', [], ...
    'isOutlet', [], ...
    'isSolLiq', [], ...
    'isConnBound', []);

connectBound = storestruct(connectBound, { ...
    'isInlet' connectIsInlet
    'isOutlet' connectIsOutlet
    'isSolLiq' connectIsSolLiq
    'isConnBound' isConnBound
    } ...
);

Model.connect = connect;
Model.connectBound = connectBound;

end

% ----- %

function A = assemblematrix(Model)

fprintf('\n\nAssembling coefficient matrix...');

nodeList = sort([Model.lids.inlet; Model.lids.outlet; Model.lids.solLiq;
Model.lids.interior]);

nStencilPts = 7;
% size of finite-difference stencil
maxMatBounds = repmat(numel(nodeList), [1 2]); %
maximum matrix bounds for stiffness matrix

nNeighSolLiq = sum(Model.connectBound.isSolLiq, 2);
notC_isSolLiq = ~Model.connectBound.isSolLiq;

enterVals = [ ...
    -(nStencilPts - 1)*ones(size(Model.connect, 1), 1) + nNeighSolLiq; ...
    notC_isSolLiq(:,2); ...
    notC_isSolLiq(:,3); ...
    notC_isSolLiq(:,4); ...
    notC_isSolLiq(:,5); ...
    notC_isSolLiq(:,6); ...
    notC_isSolLiq(:,7) ...
    ];

[~, indInt] = sort(nodeList);
indMat = zeros(size(nodeList));
indMat(nodeList) = indInt;

A = sparse(repmat(indMat(Model.connect(:,1))), [7 1]), ...
    [indMat(Model.connect(:,1)); ...
    indMat(Model.connect(:,2)); ...
    indMat(Model.connect(:,3)); ...
    indMat(Model.connect(:,4)); ...
    indMat(Model.connect(:,5)); ...
    indMat(Model.connect(:,6)); ...
    indMat(Model.connect(:,7))], ...
    enterVals, ...
    maxMatBounds(1), maxMatBounds(2), numel(enterVals));

A = A(:,indMat(Model.lids.interior));
A = A.';

```

```

A = A(:,indMat(Model.lids.interior));
A = A.';

fprintf('\n\n\tNumber of degrees of freedom: %i\n',
numel(Model.lids.interior));

end

% ----- %

function b = assembleloadvector(Model)

% nConnBound = sum(Model.connectBound.isConnBound);
% number of connections pertaining to each node

npts = prod(Model.geom.dim);

% b = spalloc(npts, 1, nConnBound);
% allocating memory for sparse load vector
b = zeros(npts, 1);

% boundSum = -1*( ... %
% applying Dirchlet and Neumann boundary conditions
% Model.params.V_inlet * sum(Model.connectBound.isInlet, 2) + ...
% Model.params.V_outlet * sum(Model.connectBound.isOutlet, 2) ...
% );
%
% iNonzero = find(boundSum(
%
% b = sparse

b(Model.connect(:,1)) = -1*( ...
% applying Dirchlet and Neumann boundary conditions
% Model.params.V_inlet * sum(Model.connectBound.isInlet, 2) + ...
% Model.params.V_outlet * sum(Model.connectBound.isOutlet, 2) ...
% );

b = b(Model.lids.interior);
% removing pixels that belong to the boundary or solid material
b = sparse(b);

end

% ----- %

function parentStruct = clearstruct(parentStruct, childStruct)

nstr = size(childStruct, 1);
for istr = 1 : nstr
    parentStruct.(childStruct{istr,1}) = [];
end

end

% ----- %

function x = dcsolver(A, b, solveStruct)

switch solveStruct.soltype

    case 'direct'

        fprintf('\nDirect solver: Matlab %s', '"");

        x = A\b;

    case 'iter'

        iluStruct = struct( ...
            'type', 'ict', ...
            'droptol', solveStruct.droptol, ...
            'shape', 'lower' ...

```

```

    );

    fprintf('\nPreconditioning matrix...\n');
    fprintf('\n\tPreconditioner: Incomplete Choleski Factorization');
    fprintf('\n\tdroptol: %.1e\n', solveStruct.droptol);
    fprintf('\nInitiating solver...\n');
    fprintf('\n\tIterative solver: PCG\n\treltol: %.1e\n',
solveStruct.reltol);

    L = ichol(-1*A, iluStruct);
    [x, flag, rr1, iter, relNorm] = pcg(-1*A, -1*b, solveStruct.reltol,
solveStruct.maxiter, L, L');

    switch flag
    case 0
        fprintf('\n\tPCG converged to the desired tolerance %.1e
within %i iterations.\n', solveStruct.reltol, numel(relNorm));
    case 1
        error('\n\tPCG iterated %i times but did not converge.\n',
solveStruct.maxiter);
    case 2
        error('\n\tPreconditioner was ill-conditioned.\n');
    case 3
        error('\n\tPCG stagnated.\n');
    end

    % Model.result.flag = flag;
    % Model.result.iter = iter;
    % Model.result.relNorm = relNorm;
    % Model.result.rr1 = rr1;

    figure(1); clf;
    plot(1:numel(relNorm), relNorm, '-o');
    set(gca, 'Yscale', 'log');
    title('Convergence');
    xlabel('Iterations');
    ylabel('Relative Norm');

end

% Model.result.potential = x;

end

% ----- %

function Model = postprocessing(Model, x)

fprintf('\nPost-processing\n');

% Post-processing

v = zeros(prod(Model.geom.dim), 1);

v(Model.lids.interior) = x;
% remapping solution to 3D geometry
v(Model.lids.inlet) = Model.params.V_inlet;
v(Model.lids.outlet) = Model.params.V_outlet;

v = reshape(v, Model.geom.dim(1), Model.geom.dim(2), Model.geom.dim(3));
v = v(:,2:end-1,2:end-1);

G = Model.geom.G(:,2:end-1,2:end-1);

ex = -1 * (v(2:end, :, :) - v(1:end-1, :, :)) .* G(1:end-1, :, :). * G(2:end, :, :);
ey = -1 * (v(:, 2:end, :) - v(:, 1:end-1, :)) .* G(:, 1:end-1, :). * G(:, 2:end, :);
ez = -1 * (v(:, :, 2:end) - v(:, :, 1:end-1)) .* G(:, :, 1:end-1). * G(:, :, 2:end);

jx = Model.params.sigma * ex;
jy = Model.params.sigma * ey;
jz = Model.params.sigma * ez;

```

```

E = cell(1, 3);
J = cell(1, 3);

switch Model.params.flowAxis
    case 'X'

        rotG = Model.geom.G;

        E{1} = ex;
        E{2} = ey;
        E{3} = ez;

        J{1} = jx;
        J{2} = jy;
        J{3} = jz;

    case 'Y'

        rotG = flip(permute(Model.geom.G, [2 1 3]), 1);
        V = flip(permute(V, [2 1 3]), 1);

        E{1} = flip(permute(ey, [2 1 3]), 1);
        E{2} = flip(permute(ex, [2 1 3]), 1);
        E{3} = flip(permute(ez, [2 1 3]), 1);

        J{1} = flip(permute(jy, [2 1 3]), 1);
        J{2} = flip(permute(jx, [2 1 3]), 1);
        J{3} = flip(permute(jz, [2 1 3]), 1);

    case 'Z'

        rotG = flip(permute(Model.geom.G, [3 2 1]), 1);
        V = flip(permute(V, [3 2 1]), 1);

        E{1} = flip(permute(ez, [3 2 1]), 1);
        E{2} = flip(permute(ey, [3 2 1]), 1);
        E{3} = flip(permute(ex, [3 2 1]), 1);

        J{1} = flip(permute(jz, [3 2 1]), 1);
        J{2} = flip(permute(jy, [3 2 1]), 1);
        J{3} = flip(permute(jx, [3 2 1]), 1);

end

dVdL = -1 * (Model.params.V_outlet - Model.params.V_inlet) / (Model.geom.L - 1);

switch Model.params.flowAxis
    case 'X'
        jxAvg = (1/prod([size(G, 2), size(G, 3)])) *
sum(sum(J{1}(:,:floor(size(G, 1) / 2)))));
        sigmaEff = jxAvg / dVdL;
    case 'Y'
        jyAvg = (1/prod([size(G, 1), size(G, 3)])) *
sum(sum(J{2}(:,:floor(size(G, 2) / 2)))));
        sigmaEff = jyAvg / dVdL;
    case 'Z'
        jzAvg = (1/prod([size(G, 1), size(G, 2)])) *
sum(sum(J{3}(:,:floor(size(G, 3) / 2)))));
        sigmaEff = jzAvg / dVdL;
end

Model.result = [];
Model.result = storestruct(Model.result, { ...
    'sigmaEff' sigmaEff
    'V' V
    'E' E
    'J' J
    'rotG' rotG
});

```

```

fprintf('\nBulk electrical conductivity: %.4e\n', ...
    Model.result.sigmaEff);

End

% ----- %

function saveresult(sDir, Model, nRand)

    if ~exist(sDir, 'dir')
        mkdir(sDir);
    end

    V = Model.result.V;

    E = cellfun(@single, Model.result.E, 'UniformOutput', false);
    J = cellfun(@single, Model.result.J, 'UniformOutput', false);

    ex = E{1};
    ey = E{2};
    ez = E{3};

    jx = J{1};
    jy = J{2};
    jz = J{3};

    sigmaStr = sprintf('%.3f-', Model.params.sigma(:)); sigmaStr =
sigmaStr(1:end-1);

    newDir = sprintf('%sflow%s_sigma%s_%04i/', sDir, Model.params.flowAxis,
sigmaStr, nRand);
    if exist(newDir, 'dir') == 0
        mkdir(newDir);
    end

    print(1, '-dpng', [newDir, 'convergence']);

    save(sprintf('%sstruct', newDir), 'Model', '-v7.3');
    fprintf('\nSaving %sstruct.mat', newDir);
    save(sprintf('%spotential', newDir), 'V', '-v7.3');
    fprintf('\nSaving %spotential.mat', newDir);
    save(sprintf('%selectricField', newDir), 'E', '-v7.3');
    fprintf('\nSaving %selectricField.mat', newDir);
    save(sprintf('%scurrentDensity', newDir), 'J', '-v7.3');
    fprintf('\nSaving %scurrentDensity.mat', newDir);

    fprintf('\n');
    % save(sprintf('%s_struct', newDir, newDir, 'ex', '-v7.3');
    % fprintf('\nSaving %sflow%s_refine%i_sigma%s_electricFieldX.mat',
newDir);
    % save(sprintf('%s_struct', newDir, newDir, 'ey', '-v7.3');
    % fprintf('\nSaving %sflow%s_refine%i_sigma%s_electricFieldY.mat',
newDir);
    % save(sprintf('%s_struct', newDir, newDir, 'ez', '-v7.3');
    % fprintf('\nSaving %sflow%s_refine%i_sigma%s_electricFieldZ.mat',
newDir);

    % save(sprintf('%s_struct', newDir, newDir, 'jx', '-v7.3');
    % fprintf('\nSaving %s_currentDensityX.mat', newDir);
    % save(sprintf('%s_struct', newDir, newDir, 'jy', '-v7.3');
    % fprintf('\nSaving %s_currentDensityY.mat', newDir);
    % save(sprintf('%s_struct', newDir, newDir, 'jz', '-v7.3');
    % fprintf('\nSaving %s_currentDensityZ.mat', newDir);

end

```

Appendix C: Summary of experimental charges and methods for measuring local melt fraction distribution

C.1 Summary of harzburgite samples

See next page.

Sample	^a Suite	^b Starting Comp.	^c Experimenter	Nominal Melt Fraction	Sintering Duration [hours]	Successful Exp. Run	^d Core Orientation	^e Analytical Procedures	Beam Energy	Notes
hzb-1	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.20	168	No	N/A	None		
hzb-2	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.10	168	Yes	perpendicular	Tomo		
hzb-3	regular / time series	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.05	168	Yes	perpendicular	Tomo	24.4	High melt fraction (~0.20) & no opx (opx must have dissolved)
hzb-4	time series	(82-Ol / 18-Opx) + 8272f-Ba	Le Roux	0.05	336	No	parallel	Quant. Chem., EDS, tomo	24.4	strange texture and Ca-rich melt
hzb-5	time series	(82-Ol / 18-Opx) + 8272f-Ba	Le Roux	0.20	168	Yes	parallel	Quant. Chem., EDS, tomo	24.4	
hzb-6	time series	(82-Ol / 18-Opx) + 8272f-Ba	Le Roux	0.05	42	Yes	parallel	Quant. Chem., EDS, tomo	24.4	
hzb-7	time series	(82-Ol / 18-Opx) + 8272f-Ba	Le Roux	0.05	84	Yes	parallel	Quant. Chem., EDS, tomo	24.4	
hzb-8	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.05		No	N/A	N/A		Early termination
hzb-9	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.05	168	Yes	parallel	Tomo / WHOI	24.4	bubbles
hzb-10	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.05	336	Yes	parallel	SEM	24.4	bubbles
hzb-11	regular	(82-Ol / 18-Opx) + 8272f-Ba	Miller	0.02	336	Yes	parallel	Tomo / WHOI	24.4	bubbles
hzb-12	regular	(60-Ol / 40-Opx) + 8272f-Ba	Miller	0.10	200	Yes	parallel	SEM	24.4	
hzb-13	regular	(60-Ol / 40-Opx) + 8272f-Ba	Miller	0.02	336	Yes	parallel	Tomo / WHOI	24.4	fine grained
hzb-14	regular	(60-Ol / 40-Opx) + 8272f-Ba	Miller	0.20	200	Yes	parallel	SEM	24.4	
hzb-15	regular	(60-Ol / 40-Opx) + 8272f-Ba	Miller	0.05	168	Yes	parallel	Tomo	24.4	

Table C.1: Summary of samples composed of olivine-opx-basaltic melt. ^aTwo suites of experiments were conducted for the harzburgite samples: (1) melt fraction varied (regular), (2) sintering time varies (time-series). ^bStarting composition. (x-Ol / y-Opx) is an oxide mix that produces x to y volume proportion olivine to opx. 8272f-Ba is the identification number of the basalt, which comes from Medicine Lake Volcano. ^cName of experimenter that prepared the sample. ^dCores were drilled from samples either parallel or perpendicular to the cylindrical axis. ^eTypes of analysis: tomography (tomo), quantitative chemistry (quant. chem.), electron dispersive spectroscopy (EDS), or scanning electron microscope (SEM).

C.2 Quantitative chemistry analysis for harzburgite samples

See next page.

Table C.2: Quantitative chemical analysis conducted for harzburgite samples conducted by Dr. Phil Piccoli. Green and orange fills signify olivine and orthopyroxene measurements, respectively.

No.	FeO	CaO	Na ₂ O	MnO	TiO ₂	MgO	NiO	Cr ₂ O ₃	SiO ₂	Al ₂ O ₃	Total	Notes
1	7.5585	0.1804	0.0133	0.0907	0	51.8744	0.0735	0.0122	41.222	0.0207	101.0456	hzb4_olivine_grain1 (light color)
2	5.1135	0.4637	0	0.0703	0.0057	53.4041	0	0.0083	41.4852	0.1239	100.6746	hzb4_olivine_grain1 (dark color)
3	7.4356	0.1731	0.0214	0.1144	0	51.7615	0.0675	0.0094	41.264	0.0187	100.8655	hzb4_olivine_grain2 (light color)
4	7.241	0.1971	0.0025	0.0842	0	51.7548	0.0605	0	41.0881	0.0275	100.4556	hzb4_olivine_grain3 (light color)
5	5.3316	0.4429	0	0.0866	0	53.0975	0.0534	0	41.1466	0.0953	100.2538	hzb4_olivine_grain3 (dark color)
6	6.9883	0.363	0.0106	0.0813	0	51.955	0.065	0	41.3994	0.1099	100.9724	hzb4_olivine_grain4 (light color)
7	5.1013	0.439	0	0.0433	0	53.0172	0.0232	0.0026	41.5749	0.1352	100.3367	hzb4_olivine_grain4 (dark color)
8	7.6829	0.1305	0.0006	0.096	0	51.5529	0.148	0.0204	40.8455	0.056	100.5327	hzb4_olivine_grain5 (light color)
9	5.152	0.4473	0.0006	0.0614	0	53.1614	0.0753	0.0197	41.3981	0.1322	100.4479	hzb4_olivine_grain5 (dark color)
10	6.9331	0.17	0.015	0.1061	0	51.7996	0.0491	0.0102	41.2566	0.0607	100.4003	hzb4_olivine_grain6 (light color)
11	5.0139	0.4958	0	0.0426	0	53.3991	0	0	41.4087	0.1418	100.5018	hzb4_olivine_grain6 (dark color)
12	9.2665	0.1874	0.0026	0.0961	0	50.1047	0.1993	0.0028	40.5404	0.1645	100.5642	hzb5_olivine_grain1
13	9.2655	0.1675	0.0013	0.1582	0	50.1134	0.1786	0	41.005	0.1183	101.0077	hzb5_olivine_grain2
14	9.1059	0.1667	0.0083	0.1118	0	50.5291	0.2436	0.0095	40.9337	0.1027	101.2112	hzb5_olivine_grain3
15	9.3587	0.1851	0.0013	0.1011	0	50.1812	0.1868	0.0233	40.9556	0.088	101.081	hzb5_olivine_grain4
16	9.3522	0.1818	0	0.127	0	49.9992	0.2209	0	40.4618	0.2182	100.561	hzb5_olivine_grain5
17	5.6383	0.9353	0.0177	0.1328	0.0344	34.9755	0.1089	0	56.3163	1.7202	99.8794	hzb5_opx_grain1
18	5.8887	0.7202	0.031	0.1089	0.0057	35.2561	0.1521	0.0321	54.9831	2.8167	99.9947	hzb5_opx_grain2
19	5.7679	0.8002	0.048	0.1123	0	34.0395	0.1471	0.006	56.4596	2.7882	100.1687	hzb5_opx_grain3
20	5.6673	0.8445	0.0189	0.1259	0.018	34.1054	0.1158	0	55.6729	2.083	98.6517	hzb5_opx_grain4
21	5.389	0.9278	0.0239	0.1129	0.0063	34.1861	0.1335	0	56.8367	2.3469	99.9632	hzb5_opx_grain5
22	8.9372	0.1264	0	0.0626	0	49.754	0.3272	0	40.7945	0.0762	100.0781	hzb6_olivine_grain1
23	8.9607	0.1156	0.009	0.1275	0	49.8777	0.2844	0	40.7927	0.166	100.3335	hzb6_olivine_grain2
24	8.9812	0.1329	0	0.1168	0.0017	49.4154	0.3239	0	40.6141	0.3803	99.9664	hzb6_olivine_grain3
25	8.9816	0.1131	0.0212	0.1422	0.0043	49.7685	0.3206	0.0095	40.5346	0.0811	99.9768	hzb6_olivine_grain4
26	8.9625	0.1033	0	0.1179	0.0073	49.7172	0.3128	0	40.7353	0.1318	100.088	hzb6_olivine_grain5
27	8.9528	0.1008	0	0.1083	0	49.7378	0.2709	0	41.0818	0.157	100.4093	hzb7_olivine_grain1

28	8.8917	0.1558	0	0.1361	0	49.9958	0.2599	0	40.8594	0.0684	100.367	hzb7_olivine_grain2
29	8.9337	0.1207	0	0.1349	0.0007	50.0392	0.254	0	41.2859	0.0965	100.8655	hzb7_olivine_grain3
30	8.8278	0.133	0	0.0931	0	50.261	0.2154	0	41.25	0.1551	100.9353	hzb7_olivine_grain4
31	7.9342	0.3487	0	0.1045	0	45.2715	0.2125	0	44.6675	0.8602	99.3991	hzb7_olivine_grain5
32	5.5126	0.7869	0.0107	0.1304	0	34.3511	0.0321	0	56.3012	3.1837	100.3086	hzb6_opx_grain1
33	5.6289	0.9134	0.0164	0.1032	0.001	34.1612	0.105	0	57.0786	2.4104	100.4181	hzb6_opx_grain2
34	5.7628	0.8291	0.0246	0.1087	0.04	34.7259	0.0864	0	55.9997	2.8054	100.3825	hzb6_opx_grain3
35	5.876	0.7282	0	0.1024	0.0183	34.087	0.1296	0	56.6388	3.0596	100.6399	hzb6_opx_grain4
36	5.7031	0.827	0	0.1104	0.0113	35.2218	0.1296	0.002	56.7949	2.692	101.4921	hzb6_opx_grain5
37	7.3388	0.1952	0	0.1118	0	51.3023	0.0834	0	41.2113	0.0546	100.2973	hzb7_olivine_grain1
38	5.0616	0.4292	0.0218	0.0617	0	53.2406	0.0378	0.0043	41.643	0.1343	100.6342	hzb7_olivine_grain2
39	9.1212	0.1921	0	0.1374	0.0185	50.0733	0.2293	0	40.9718	0.0926	100.8362	hzb7_olivine_grain3
40	8.8478	2.2261	0.242	0.1373	0.0281	41.9597	0.2729	0	41.8196	3.7575	99.2911	hzb7_olivine_grain4
41	8.9042	0.1304	0	0.103	0.001	49.8999	0.257	0.0229	40.9286	0.0642	100.3111	hzb7_olivine_grain5
46	5.7994	1.2445	0.0654	0.0954	0.0296	33.5607	0.0816	0	56.3094	5.0146	102.2005	hzb7_opx_grain1
47	5.7342	1.353	0.0282	0.1054	0.0259	33.4651	0.0586	0	56.0744	4.4326	101.2774	hzb7_opx_grain2
48	5.7547	1.2659	0.0653	0.098	0	33.8405	0.0884	0	56.7424	3.5035	101.3587	hzb7_opx_grain3
49	5.7109	1.4412	0.049	0.1486	0.0451	33.7973	0.1231	0.041	56.248	4.161	101.7652	hzb7_opx_grain4
50	5.8557	1.2874	0.0589	0.1103	0.0089	34.4247	0.0707	0	56.1995	3.5829	101.599	hzb7_opx_grain5

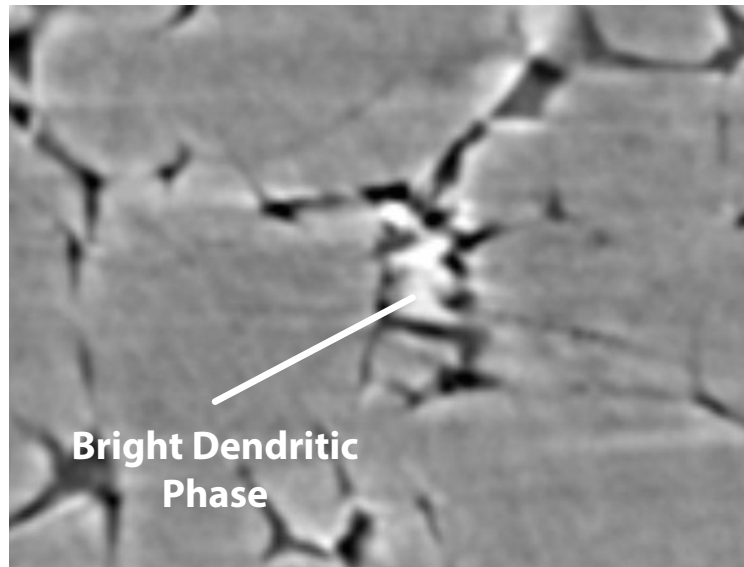


Figure C.1: Bright dendritic phase appears to be partially crystallized basalt and is assumed to be melt during segmentation, since they should not be present at run conditions.

```

% ----- %

function [phiOl, phiOpx, Stats] = LPAnalyze(inputFile, p, varargin)
% Characterizes the local melt fraction distribution of a label image.
% Requires melt, olivine, and opx label images as input. Also requires
% separated olivine and opx 16-bit grain label images and .txt document
% containing the centers of each grain.

if ~isempty(varargin)
    printSwitch = ~isempty(find(strcmp('Print', varargin), 1));
    writeSwitch = ~isempty(find(strcmp('Write', varargin), 1));
    plotSwitch = ~isempty(find(strcmp('Plot', varargin), 1));
    saveSwitch = ~isempty(find(strcmp('Save', varargin), 1));
    plotGrain = ~isempty(find(strcmp('Plot Grain', varargin), 1));
else
    printSwitch = 0;
    writeSwitch = 0;
    plotSwitch = 0;
    saveSwitch = 0;
    plotGrain = 0;
end

% -----
% Reading the melt and grain files. Note: Be sure that the input grain
% files have had grains intersecting the bounding box removed.
% -----
fprintf('-----\n');
warning('off', 'all');

listDir = 'C:\Users\kevinmiller\code\LPAnalyze_new\'; % directory for binary
tif images

FileNames = LabelFileReader(listDir, inputFile);

nfile = length(FileNames.DirTif);
for ifile = 1 : nfile
    % Loading the binary files
    fprintf('\nReading %s...', FileNames.LabelOlName{ifile});
    LabelOlTif = Tif3DReader(FileNames.DirTif{ifile},
FileNames.LabelOlName{ifile});
    fprintf('Completed!\n');
    fprintf('Reading %s...', FileNames.BinOlName{ifile});
    BinOlTif = Tif3DReader(FileNames.DirTif{ifile},
FileNames.BinOlName{ifile});
    fprintf('Completed!\n');
    fprintf('Reading %s...', FileNames.LabelOpxName{ifile});
    LabelOpxTif = Tif3DReader(FileNames.DirTif{ifile},
FileNames.LabelOpxName{ifile});
    fprintf('Completed!\n');
    fprintf('Reading %s...', FileNames.BinOpxName{ifile});
    BinOpxTif = Tif3DReader(FileNames.DirTif{ifile},
FileNames.BinOpxName{ifile});
    fprintf('Completed!\n');
    fprintf('Reading %s...', FileNames.BinMeltName{ifile});
    BinMeltTif = Tif3DReader(FileNames.DirTif{ifile},
FileNames.BinMeltName{ifile});
    fprintf('Completed!\n');

    % Loading the quantitative grain analyses
    DirAnl = 'C:\Users\kevinmiller\data\analysis\'; % directory for the
quantitative grain analyses
    fidOl = fopen([DirAnl, FileNames.OlAnlName{ifile}]);
    fidOpx = fopen([DirAnl, FileNames.OpxAnlName{ifile}]);
    GrainAnlOl = textscan(fidOl, '%d %d %d %d', 'HeaderLines', 1);
    fprintf('\nReading %s\n', FileNames.OlAnlName{ifile});
    GrainAnlOpx = textscan(fidOpx, '%d %d %d %d', 'HeaderLines', 1);
    fprintf('Reading %s\n', FileNames.OpxAnlName{ifile});
    fclose(fidOl); fclose(fidOpx); % closing the file identifiers

    nol = max(max(max(LabelOlTif))); % number of olivine grains

```

```

nOpx = max(max(max(LabelOpxTif))); % number of opx grains

olCtr = double([GrainAnOl{2}, GrainAnOl{3}, GrainAnOl{4}]) + 1; %
centers of olivine grains
OpxCtr = double([GrainAnOpx{2}, GrainAnOpx{3}, GrainAnOpx{4}]) + 1; %
centers of opx grains

% allocating memory for the list of local melt fractions for each phase
phiOl = nan(nOl, 1);
phiOpx = nan(nOpx, 1);

usInd = strfind(FileNames.BinMeltName{ifile}, '_');
dotInd = strfind(FileNames.BinMeltName{ifile}, '.');
xInd = strfind(FileNames.BinMeltName{ifile}(usInd(3)+1:usInd(4)-1),
'x');
dim = [ ... % determining the dimensions of the subvolume from the name
of BinMeltName
str2double(FileNames.BinMeltName{ifile}(usInd(3)+1:usInd(3)+xInd(1)-
1)), ...

str2double(FileNames.BinMeltName{ifile}(usInd(3)+xInd(1)+1:usInd(3)+xInd(2)-
1)), ...
str2double(FileNames.BinMeltName{ifile}(usInd(3)+xInd(2)+1:usInd(4)-
1))];
seriesID = sprintf('%s', FileNames.BinMeltName{ifile}(1:usInd(1)-1));
sampleID = sprintf('%s',
FileNames.BinMeltName{ifile}(usInd(1)+1:usInd(2)-1));
subvolID = sprintf('%s',
FileNames.BinMeltName{ifile}(usInd(4)+1:dotInd(1)-1));
anlName = sprintf('%s_%s_%ix%i%i_%s.LPAnalysis_p%.2f', seriesID,
sampleID, dim(1), dim(2), dim(3), subvolID, p);

fprintf('\nSubvolume dimensions: %ix%i%i pixels^3\n', dim(1), dim(2),
dim(3));

% -----
--
% Doing a loop for olivine grains
% -----
--
fprintf('\nAnalyzing olivine grains\n');
ismore = 1;
for iOl = 1 : nOl
    iOlCtr = olCtr(iOl,:);
    [ii, jj, kk] = ind2sub(size(LabelOlTif), find(LabelOlTif==iOl)); %
finding the location of each pixel belonging to grain iOl
    % isPlane = numel(unique(kk)) == 1;
    if ~(numel(unique(ii)) == 1 || numel(unique(jj)) == 1 ||
numel(unique(kk)) == 1)
        % k = convhull(ii, jj, kk); % reducing the number of
points to a simplified convex hull
        k = convhull(ii, jj, kk, 'simplify', false); % reducing the
number of points to a simplified convex hull
        hullpts = [jj(k(:,2)), ii(k(:,1)), kk(k(:,3))]; % combining the
hull points into an array
        T0 = [1 0 0 -iOlCtr(1); 0 1 0 -iOlCtr(2); 0 0 1 -iOlCtr(3); 0 0
0 1];
        hullpts0 = T0*[hullpts'; ones(1, size(hullpts, 1))]; hullpts0 =
hullpts0(1:3,:); % translating the hullpts to the origin
        hullpts0 = [hullpts0(:,2), hullpts0(:,1), hullpts0(:,3)];

        [~, radii0, Pevects, ~] = ellipsoid_fit(hullpts0); % fitting the
convex hull points to an ellipsoid
        if any(isnan(radii0))
            radii0 = sqrt(-1);
        end
    else
        radii0 = sqrt(-1);
    end
end

```

```

        if isreal(radII0) || ~any(isnan(radII0)) % checking that radII is
        real, since ellipsoid_fit can return imaginary values if hullpts0 is noisy

        radII0New = p*radII0; % calculating the new ellipsoid parameters
        based on the dialation parameter, p
        DPNew = diag(radII0New.^-2); % diagonalizing the principal
        lengths
        PNew = Pevecs*DPNew*Pevecs'; % rotating back to the grain's
        reference

        parsNew = [PNew(1,1); PNew(2,2); PNew(3,3); PNew(1,2);
        PNew(1,3); PNew(2,3)]; % list of the new paramters of the dialated ellipsoid

        xmin = -sqrt(1/(sign(parsNew(1))*parsNew(1)));
        xmax = sqrt(1/(sign(parsNew(1))*parsNew(1)));

        ymin = -sqrt(1/(sign(parsNew(2))*parsNew(2)));
        ymax = sqrt(1/(sign(parsNew(2))*parsNew(2)));

        zmin = -sqrt(1/(sign(parsNew(3))*parsNew(3)));
        zmax = sqrt(1/(sign(parsNew(3))*parsNew(3)));

        ellipBound = [ ... % coordinates for box bounding the ellipsoid
        sign(xmin)*(ceil(abs(xmin)) + 1),
        sign(xmax)*(ceil(abs(xmax)) + 1); ...
        sign(ymin)*(ceil(abs(ymin)) + 1),
        sign(ymax)*(ceil(abs(ymax)) + 1); ...
        sign(zmin)*(ceil(abs(zmin)) + 1),
        sign(zmax)*(ceil(abs(zmax)) + 1)];

        % translating to the center of the grain
        T2 = [1 0 0 iolCtr(2); 0 1 0 iolCtr(1); 0 0 1 iolCtr(3); 0 0 0
        1];
        ellipBoundT = T2*[ellipBound; ones(1, 2)]; ellipBoundT =
        ellipBoundT(1:3,:);

        if ~(any(ellipBoundT(1,:) < 1 | ellipBoundT(1,:) > dim(1)) ||
        ... % exclude grains that intersect the boundary of the subvolume
        any(ellipBoundT(2,:) < 1 | ellipBoundT(2,:) > dim(2)) ||
        ...
        any(ellipBoundT(3,:) < 1 | ellipBoundT(3,:) > dim(3)))

        meltBound = BinMeltTif(ellipBoundT(1,1) : ellipBoundT(1,2),
        ellipBoundT(2,1) : ellipBoundT(2,2), ellipBoundT(3,1) : ellipBoundT(3,2));

        % indexing the binary melt image and translating it to the
        origin
        [iAll, jAll, kAll] = ind2sub(size(meltBound),
        find(meltBound==1 | meltBound==0)); % finding the xyz coordinates of all
        pixels in the cropped melt image
        [iMelt, jMelt, kMelt] = ind2sub(size(meltBound),
        find(meltBound==1)); % finding the xyz coordinates of pixels associated with
        melt in the cropped melt image

        ctrLocal = [floor((max(iAll) - min(iAll))/2),
        floor((max(jAll) - min(jAll))/2), floor((max(kAll) - min(kAll))/2)] + 1; %
        center of the sample region where the corner is on the origin

        T3 = [1 0 0 -ctrLocal(1); 0 1 0 -ctrLocal(2); 0 0 1 -
        ctrLocal(3); 0 0 0 1]; % assembling translation matrix for translating to
        the origin

        ijkAll = [iAll, jAll, kAll]; % concatenating all pixel
        coordinates
        ijkAllT = T3*[ijkAll'; ones(1, numel(iAll))]; ijkAllT =
        ijkAllT(1:3,:); % translating to the origin
        [inptsAll, ~, ~] = inoutEllipGen2(ijkAllT, parsNew);

        ijkMelt = [iMelt, jMelt, kMelt]; % concatenating melt pixel
        coordinates
        ijkMeltT = T3*[ijkMelt'; ones(1, numel(iMelt))]; ijkMeltT =
        ijkMeltT(1:3,:); % translating back to the original cropped melt indices

```

```

[inptsMelt, ~, ~] = inoutEllipGen2(ijkMeltT, parsNew);
if plotGrain
    figure(5); clf; hold on;
    % plot3(ijkMeltT(ijkMeltT(:,2) > 0,2), ijkMeltT(ijkMeltT(:,2) > 0,3), 'or');
    % plot3(hullpts0(:,1),
    hullpts0(:,2), hullpts0(:,3), 'o', ...
    % 'MarkerSize', 12, ...
    % 'MarkerFaceColor', 'b');
    plot3(hullpts0(:,1), hullpts0(:,2), hullpts0(:,3), 'o',
    ...
    % 'MarkerSize', 12, ...
    % 'MarkerFaceColor', 'b');
    % plot3(inptsMelt(inptsMelt(:,2) >
    0,1), inptsMelt(inptsMelt(:,2) > 0,2), inptsMelt(inptsMelt(:,2) > 0,3),
    'oc', ...
    % 'MarkerSize', 12, ...
    % 'MarkerFaceColor', 'c');
    plot3(inptsMelt(:,1), inptsMelt(:,2), inptsMelt(:,3),
    'o', ...
    % 'MarkerSize', 10, ...
    % 'MarkerFaceColor', 'g');
    % F =
    Pevecs*diag(radII0New)*Pevecs';
    % [XS, YS, ZS] = sphere(100);
    % XYZe = F*[XS(:)'; YS(:)'; ZS(:)'];
    XYZe = XYZe(1:3,:);
    % Xe = reshape(XYZe(:,1), size(XS,
    1), size(XS, 2));
    % Ye = reshape(XYZe(:,2), size(YS,
    1), size(YS, 2));
    % Ze = reshape(XYZe(:,3), size(ZS,
    1), size(ZS, 2));
    % deform = (Xe.^2 + Ye.^2 +
    Ze.^2).^0.5;
    % s1 = surf(Xe, Ye, Ze, deform);
    % set(s1, ...
    % 'FaceColor', 'none');
    axis equal tight;
    box on;
    view(-30, 30);
end

nAll = size(inptsAll, 1); % number of pixels bounded by
ellipsoid
nMelt = size(inptsMelt, 1); % number of pixels associated
with melt inside the boundary ellipsoid

iphiOl = nMelt/nAll; % melt fraction for current region
phiOl(ismore,1) = iphiOl; % storing the local melt fraction
ismore = ismore + 1; % moving on to the next grain
fprintf('\t%i / %i olivine grains analyzed; Local melt
fraction: %.4f\n', iOl, nOl, iphiOl); % printing progress
else
    fprintf('\t%i / %i olivine grains analyzed; Local melt
fraction: Out of Bounds\n', iOl, nOl);
end
else
    fprintf('\t%i / %i olivine grains analyzed; Local melt fraction:
Radii are imaginary\n', iOl, nOl);
end
end

% -----
--
% Doing a loop for opx grains
% -----
--
fprintf('\nAnalyzing opx grains\n');
ismore = 1;

```



```

for iOpX = 1 : nOpX
    if iOpX == 73
        disp('');
    end
    iOpXctr = OpXctr(iOpX,:);
    [ii, jj, kk] = ind2sub(size(LabelOpXTif), find(LabelOpXTif==iOpX));
% finding the location of each pixel belonging to grain iOpX
    if ~(numel(unique(ii)) == 1 || numel(unique(jj)) == 1 ||
numel(unique(kk)) == 1)
        % k = convhull(ii, jj, kk); % reducing the number of
points to a simplified convex hull
        k = convhull(ii, jj, kk, 'simplify', false); % reducing the
number of points to a simplified convex hull
        hullpts = [jj(k(:,2)), ii(k(:,1)), kk(k(:,3))]; % combining the
hull points into an array
        T0 = [1 0 0 -iOpXctr(1); 0 1 0 -iOpXctr(2); 0 0 1 -iOpXctr(3); 0
0 0 1];
        hullpts0 = T0*[hullpts'; ones(1, size(hullpts, 1))]; hullpts0 =
hullpts0(1:3,:); % translating the hullpts to the origin
        hullpts0 = [hullpts0(:,2), hullpts0(:,1), hullpts0(:,3)];

        [~, radii0, Pevecs, ~] = ellipsoid_fit(hullpts0); % fitting the
convex hull points to an ellipsoid
    else
        radii0 = sqrt(-1);
    end

    if isreal(radii0) && ~any(isnan(radii0)) % checking that radii is
real, since ellipsoid_fit can return imaginary values if hullpts0 is noisy
        radii0New = p*radii0; % calculating the new ellipsoid parameters
based on the dialation parameter, p
        DPNew = diag(radii0New.^-2); % diagonalizing the principal
lengths
        PNew = Pevecs*DPNew*Pevecs'; % rotating back to the grain's
reference

        parsNew = [PNew(1,1); PNew(2,2); PNew(3,3); PNew(1,2);
PNew(1,3); PNew(2,3)]; % list of the new paramters of the dialated ellipsoid

        xmin = -sqrt(1/(sign(parsNew(1))*parsNew(1)));
        xmax = sqrt(1/(sign(parsNew(1))*parsNew(1)));

        ymin = -sqrt(1/(sign(parsNew(2))*parsNew(2)));
        ymax = sqrt(1/(sign(parsNew(2))*parsNew(2)));

        zmin = -sqrt(1/(sign(parsNew(3))*parsNew(3)));
        zmax = sqrt(1/(sign(parsNew(3))*parsNew(3)));

        ellipBound = [ ... % coordinates for box bounding the ellipsoid
sign(xmin)*(ceil(abs(xmin)) + 1),
sign(xmax)*(ceil(abs(xmax)) + 1); ...
sign(ymin)*(ceil(abs(ymin)) + 1),
sign(ymax)*(ceil(abs(ymax)) + 1); ...
sign(zmin)*(ceil(abs(zmin)) + 1),
sign(zmax)*(ceil(abs(zmax)) + 1)];

        % translating to the center of the grain
        T2 = [1 0 0 iOpXctr(2); 0 1 0 iOpXctr(1); 0 0 1 iOpXctr(3); 0 0
0 1];
        ellipBoundT = T2*[ellipBound; ones(1, 2)]; ellipBoundT =
ellipBoundT(1:3,:);

        if ~(any(ellipBoundT(1,:) < 1 | ellipBoundT(1,:) > dim(1)) ||
... % exclude grains that intersect the boundary of the subvolume
any(ellipBoundT(2,:) < 1 | ellipBoundT(2,:) > dim(2)) ||
...
any(ellipBoundT(3,:) < 1 | ellipBoundT(3,:) > dim(3)))

        meltBound = BinMeltTif(ellipBoundT(1,1) : ellipBoundT(1,2),
ellipBoundT(2,1) : ellipBoundT(2,2), ellipBoundT(3,1) : ellipBoundT(3,2));

```

```

                                % indexing the binary melt image and translating it to the
origin
    [iAll, jAll, kAll] = ind2sub(size(meltBound),
    find(meltBound==1 | meltBound==0)); % finding the xyz coordinates of all
    pixels in the cropped melt image
    [iMelt, jMelt, kMelt] = ind2sub(size(meltBound),
    find(meltBound==1)); % finding the xyz coordinates of pixels associated with
    melt in the cropped melt image

    ctrLocal = [floor((max(iAll) - min(iAll))/2),
    floor((max(jAll) - min(jAll))/2), floor((max(kAll) - min(kAll))/2)] + 1; %
    center of the sample region where the corner is on the origin

    T3 = [1 0 0 -ctrLocal(1); 0 1 0 -ctrLocal(2); 0 0 1 -
    ctrLocal(3); 0 0 0 1]; % assembling translation matrix for translating to
    the origin

    ijkAll = [iAll, jAll, kAll]; % concatenating all pixel
coordinates
    ijkAllT = T3*[ijkAll'; ones(1, numel(iAll))]; ijkAllT =
    ijkAllT(1:3,:); % translating to the origin
    [inptsAll, ~, ~] = inoutEllipGen2(ijkAllT, parsNew);

    ijkMelt = [iMelt, jMelt, kMelt]; % concatenating melt pixel
coordinates
    ijkMeltT = T3*[ijkMelt'; ones(1, numel(iMelt))]; ijkMeltT =
    ijkMeltT(1:3,:); % translating back to the original cropped melt indices
    [inptsMelt, ~, ~] = inoutEllipGen2(ijkMeltT, parsNew);

    if plotGrain
        figure(5); clf; hold on;
        % plot3(ijkMeltT(ijkMeltT(:,2) > 0,2), ijkMeltT(ijkMeltT(:,2) > 0,3), 'or');
        % plot3(hullpts0(:,1),
        % 'MarkerSize', 12, ...
        % 'MarkerFaceColor', 'b');
        plot3(hullpts0(:,1), hullpts0(:,2), hullpts0(:,3), 'o',
        ...
        'MarkerSize', 12, ...
        'MarkerFaceColor', 'b');
        % plot3(inptsMelt(inptsMelt(:,2) >
        0,1), inptsMelt(inptsMelt(:,2) > 0,2), inptsMelt(inptsMelt(:,2) > 0,3),
        'oc', ...
        % 'MarkerSize', 12, ...
        % 'MarkerFaceColor', 'c');
        plot3(inptsMelt(:,1), inptsMelt(:,2), inptsMelt(:,3),
        'o', ...
        'MarkerSize', 10, ...
        'MarkerFaceColor', 'g');
        % F =
        Pevecs*diag(radII0New)*Pevecs';
        % [XS, YS, ZS] = sphere(100);
        % XYze = F*[XS(:)'; YS(:)'; ZS(:)'];
        XYze = XYze(1:3,:);
        % xe = reshape(XYze(:,1), size(XS,
        1), size(XS, 2));
        % ye = reshape(XYze(:,2), size(YS,
        1), size(YS, 2));
        % ze = reshape(XYze(:,3), size(ZS,
        1), size(ZS, 2));
        %
        deform = (xe.^2 + ye.^2 +
        ze.^2).^0.5;
        % s1 = surf(xe, ye, ze, deform);
        % set(s1, ...
        % 'FaceColor', 'none');
        axis equal tight;
        box on;
        view(-30, 30);
    end

```

```

nAll = size(inptsAll, 1); % number of pixels bounded by
ellipsoid
nMelt = size(inptsMelt, 1); % number of pixels associated
with melt inside the boundary ellipsoid

    iphiOpx = nMelt/nAll; % melt fraction for current region
    phiOpx(ismore,1) = iphiOpx; % storing the local melt
fraction
    ismore = ismore + 1; % moving on to the next grain
    fprintf('\t%i / %i opx grains analyzed; Local melt fraction:
%.4f\n', iOpx, nOpx, iphiOpx); % printing progress
    else
        fprintf('\t%i / %i opx grains analyzed; Local melt fraction:
Out of Bounds\n', iOpx, nOpx);
    end
    else
        fprintf('\t%i / %i opx grains analyzed; Local melt fraction:
Radii are imaginary\n', iOpx, nOpx);
    end
end

% -----
-- % Calculating the statistics for both mineral types and dumping to file
% -----
--
phiOl(isnan(phiOl) | (phiOl == 0)) = []; % removing NaN's from phiOl
phiOpx(isnan(phiOpx) | (phiOpx == 0)) = []; % removing NaN's from phiOpx

saveDir = 'C:\Users\kevinmiller\data\lp\';

if saveSwitch % saving the local melt fractions for eachgrain
    if ~exist(saveDir, 'dir')
        mkdir(saveDir)
    end
    olSaveName = sprintf('%s%s_phiOl.mat', saveDir, anIName);
    fprintf('\n%s', olSaveName);
    opxSaveName = sprintf('%s%s_phiOpx.mat', saveDir, anIName);
    fprintf('\n%s', opxSaveName);
    save(olSaveName, 'phiOl');
    save(opxSaveName, 'phiOpx');
end

totalMeltFraction = sum(BinMeltTif(:))/(size(BinMeltTif,
1)*size(BinMeltTif, 2)*size(BinMeltTif, 3)); % calculating the total melt
fraction of the subvolume region
totalOlFraction = sum(BinOlTif(:) > 0)/(size(BinOlTif, 1)*size(BinOlTif,
2)*size(BinOlTif, 3)); % calculating the total olivine fraction of the
subvolume region
totalOpxFraction = sum(BinOpxTif(:) > 0)/(size(BinOpxTif,
1)*size(BinOpxTif, 2)*size(BinOpxTif, 3)); % calculating the total olivine
fraction of the subvolume region
totalMaterialFraction = totalMeltFraction + totalOlFraction +
totalOpxFraction;

gMeanOl = geomean(phiOl*100); % geometrix mean melt fraction around
olivine grains
gStdOl = exp(sqrt(sum(log(phiOl*100./gMeanOl).^2)/numel(phiOl))); %
geometric standard deviation of local melt fraction around olivine gains

gMeanOpx = geomean(phiOpx*100); % geometric mean melt fraction around
opx grains
gStdOpx = exp(sqrt(sum(log(phiOpx*100./gMeanOpx).^2)/numel(phiOpx))); %
geometric standard deviation of local melt fraction around opx grains

R = gMeanOl/gMeanOpx; % partitioning ratio

orderOl = sort(phiOl);
medOl = median(orderOl);
lowHalfOl = orderOl(orderOl < medOl);
medLowHalfOl = median(lowHalfOl);

```

```

highHalfOl = orderOl(orderOl > medOl);
medHighHalfOl = median(highHalfOl);

Q1Ol = medLowHalfOl;
Q2Ol = medOl;
Q3Ol = medHighHalfOl;

orderOpx = sort(phiOpx);
medOpx = median(orderOpx);
lowHalfOpx = orderOpx(orderOpx < medOpx);
medLowHalfOpx = median(lowHalfOpx);
highHalfOpx = orderOpx(orderOpx > medOpx);
medHighHalfOpx = median(highHalfOpx);

Q1Opx = medLowHalfOpx;
Q2Opx = medOpx;
Q3Opx = medHighHalfOpx;

Stats.TotalMaterial.olivine = totalOlFraction;
Stats.TotalMaterial.opx = totalOpxFraction;
Stats.TotalMaterial.melt = totalMeltFraction;

Stats.Local.olivine.nGrains = nOl;
Stats.Local.olivine.mean = gMeanOl;
Stats.Local.olivine.std = gStdOl;
Stats.Local.olivine.median = medOl;
Stats.Local.olivine.quartiles = [Q1Ol, Q2Ol, Q3Ol];

Stats.Local.opx.nGrains = nOpx;
Stats.Local.opx.mean = gMeanOpx;
Stats.Local.opx.std = gStdOpx;
Stats.Local.opx.median = medOpx;
Stats.Local.opx.quartiles = [Q1Opx, Q2Opx, Q3Opx];

Stats.R = gMeanOpx/gMeanOl;

% Outputting results to text file
sprintf('\nAttempting to write metadata to file\n\t%s\n', saveDir,
anlName);
isfile = exist(sprintf('%s.txt', saveDir, anlName), 'file');
if isfile
    sprintf('\nwarning: File %s already exists\n', anlName);
end

if writeSwitch % writing to text file
    fid = fopen(sprintf('%s.txt', saveDir, anlName), 'wt');
    fprintf(fid, 'Sample Name:\n\t%s\n',
FileNames.BinMeltName{ifile}(1:dotInd(1)-1));
    fprintf(fid, '\nTotal number of grains: %i\n\tolivine: %i\n\topx:
%i\n', nOl + nOpx, nOl, nOpx);
    fprintf(fid, '\nNumber of grains used in average: %i\n\tolivine:
%i\n\topx: %i\n', numel(phiOl) + numel(phiOpx), numel(phiOl),
numel(phiOpx));
    fprintf(fid, '\nTotal Material Fractions:\n\tolivine: %.2f%\n\topx:
%.2f%\n\tmelt: %.2f%\n\tTotal: %.2f%\n', totalOlFraction*100,
totalOpxFraction*100, totalMeltFraction*100, totalMaterialFraction*100);
    fprintf(fid, '\nmelt fraction associated with each phase:\n');
    fprintf(fid, '\tolivine: %.2f% with error (-%.2f% / +%.2f%)\n',
gMeanOl, abs(gMeanOl - gMeanOl/gStdOl), abs(gMeanOl - gMeanOpx*gStdOl));
    fprintf(fid, '\topx: %.2f% with error (-%.2f% / +%.2f%)\n\n',
gMeanOpx, abs(gMeanOpx - gMeanOpx/gStdOpx), abs(gMeanOpx -
gMeanOpx*gStdOpx));
    fprintf(fid, 'Quartiles:\n\tolivine: [%.2f%, %.2f%,
%.2f%]\n\topx: [%.2f%, %.2f%, %.2f%]\n\n', Q1Ol*100, Q2Ol*100, Q3Ol*100,
Q1Opx*100, Q2Opx*100, Q3Opx*100);
    % fprintf(fid, 'Partitioning ratio:\n\t%.2f +/- %.4f
(Olivine to Opx)\n\n', R, RStd);
    fclose(fid);
end

% Printing our results in the command window at the end of run

```

```

    fprintf('\nTotal number of grains: %i\n\tOlivine: %i\n\tOpx: %i\n', nOl
+ nOpx, nOl, nOpx);
    fprintf('\nNumber of grains used in average: %i\n\tOlivine: %i\n\tOpx:
%i\n', numel(phiOl) + numel(phiOpx), numel(phiOl), numel(phiOpx));
    fprintf('\nTotal Material Fractions:\n\tOlivine: %.2f%%\n\tOpx:
%.2f%%\n\tMelt: %.2f%%\n\tTotal: %.2f%%\n', totalOlFraction*100,
totalOpxFraction*100, totalMeltFraction*100, totalMaterialFraction*100);
    fprintf('\nMelt fraction associated with each phase:\n');
    fprintf('\tOlivine: %.2f%% with error (-%.2f%% / +%.2f%%)\n', gMeanOl,
abs(gMeanOl - gMeanOl/gStdOl), abs(gMeanOl - gMeanOpx*gStdOl));
    fprintf('\tOpx: %.2f%% with error (-%.2f%% / +%.2f%%)\n\n', gMeanOpx,
abs(gMeanOpx - gMeanOpx/gStdOpx), abs(gMeanOpx - gMeanOpx*gStdOpx));
    fprintf('Quartiles:\n\tOlivine: [%.2f%%, %.2f%%, %.2f%%]\n\tOpx:
[%.2f%%, %.2f%%, %.2f%%]\n\n', Q1Ol*100, Q2Ol*100, Q3Ol*100, Q1Opx*100,
Q2Opx*100, Q3Opx*100);
    fprintf('Partitioning ratio: %.2f +/- %.4f\n\n', R);

% -----
--
% Plotting the results
% -----
--
    if plotSwitch % if 'Plot' is specified in the variable input
        figure(1); clf; subplot(211)
        nedge = 10;
        edges = linspace(0, 0.4, nedge);
        dataOl = histc(phiOl, edges);
        dataOpx = histc(phiOpx, edges);
        plot(edges, dataOl, 'g'); hold on;
        plot(edges, dataOpx, 'r'); hold off;
        xlabel('Local Melt Fraction');
        ylabel('# of Grains');

        title(sprintf('LP Histograms for %s%-s with p = %.1f and nedge =
%i', seriesID, sampleID, subvolID, p, nedge));

        XLim = get(gca, 'XLim');
        YLim = get(gca, 'YLim');

        %           text(XLim(2)*.75, YLim(2)*.6,
        sprintf('Olivine:\nMean_\phi: %.2f%%\n\sigma_\phi:
%.2f%%\nOpx:\nMean_\phi: %.2f%%\n\sigma_\phi: %.2f%%', gMeanOl, gStdOl,
gMeanOpx, gStdOpx));
        %           text(XLim(2)*.75, YLim(2)*.6, sprintf('Olivine:\nMedian_\phi:
%.2f%%\nOpx:\nMedian_\phi: %.2f%%', 100*Q2Ol, 100*Q2Opx));
        subplot(212)
        plot(sort(phiOl), linspace(0,1,numel(phiOl)), 'g');
        hold on;
        plot(sort(phiOpx), linspace(0,1,numel(phiOpx)), 'r');

        plot([Q1Ol, Q1Opx], [0 1], '--g');
        plot([Q2Ol, Q2Opx], [0 1], '--g');
        plot([Q3Ol, Q3Opx], [0 1], '--g');

        plot([Q1Opx, Q1Opx], [0 1], '--r');
        plot([Q2Opx, Q2Opx], [0 1], '--r');
        plot([Q3Opx, Q3Opx], [0 1], '--r');

        xlabel('Local Melt Fraction');
        ylabel('Cumulative frequency');
        legend('Olivine', 'Opx', 'Location', 'Southeast');
        set(gca, 'xscale', 'log');
        xlim([0.01,1]);

        if printSwitch % Saving the figure
            dotLoc = strfind(FileNames.BinMeltName{ifile}, '.');
            saveName = sprintf('%s%-s_LithPart_p%.2f.pdf', saveDir,
FileNames.BinMeltName{ifile}(1:dotLoc(1)-1), p);
            print(1, '-dpdf', saveName);
            fprintf('Saving file to:\n\t%s\n\n', saveName);
        end
    end
end

```

```

end
end

% ----- %

function FileNames = LabelFileReader(Dir, fileList)

FileNames = struct( ...
    'DirTif', {}, ...
    'BinMeltName', {}, ...
    'BinOlName', {}, ...
    'BinOpxName', {}, ...
    'LabelOlName', {}, ...
    'LabelOpxName', {}, ...
    'OlAnlName', {}, ...
    'OpxAnlName', {} ...
);

fid = fopen(sprintf('%s%s', Dir, fileList));
GoOn0 = 1;
while GoOn0
    cline = fgetl(fid);
    switch cline
        case '# Binary Files Folder'
            DirTif = {};
            GoOn1 = 1; next = 1;
            nextLine = fgetl(fid);
            while GoOn1
                if isempty(nextLine) || ~ischar(nextLine); break;
                else
                    DirTif{next,1} = nextLine;
                    next = next + 1;
                    nextLine = fgetl(fid);
                end
            end
            FileNames(1).DirTif = DirTif;
        case '# Melt 8-bit Binary File'
            BinMeltName = {};
            GoOn1 = 1; next = 1;
            nextLine = fgetl(fid);
            while GoOn1
                if isempty(nextLine) || ~ischar(nextLine); break;
                else
                    BinMeltName{next,1} = nextLine;
                    next = next + 1;
                    nextLine = fgetl(fid);
                end
            end
            FileNames.BinMeltName = BinMeltName;
        case '# Olivine 8-bit Binary File'
            BinOlName = {};
            GoOn1 = 1; next = 1;
            nextLine = fgetl(fid);
            while GoOn1
                if isempty(nextLine) || ~ischar(nextLine); break;
                else
                    BinOlName{next,1} = nextLine;
                    next = next + 1;
                    nextLine = fgetl(fid);
                end
            end
            FileNames.BinOlName = BinOlName;
        case '# Opx 8-bit Binary File'
            BinOpxName = {};
            GoOn1 = 1; next = 1;
            nextLine = fgetl(fid);
            while GoOn1
                if isempty(nextLine) || ~ischar(nextLine); break;
                else

```

```

        BinOpxName{next,1} = nextLine;
        next = next + 1;
        nextLine = fgetl(fid);
    end
end
FileNames.BinOpxName = BinOpxName;
case '# Olivine 16-bit Binary File for Interior Grains'
    LabelOlName = {};
    GoOn1 = 1; next = 1;
    nextLine = fgetl(fid);
    while GoOn1
        if isempty(nextLine) || ~ischar(nextLine); break;
        else
            LabelOlName{next,1} = nextLine;
            next = next + 1;
            nextLine = fgetl(fid);
        end
    end
    FileNames.LabelOlName = LabelOlName;
case '# Opx 16-bit Binary File for Interior Grains'
    LabelOpxName = {};
    GoOn1 = 1; next = 1;
    nextLine = fgetl(fid);
    while GoOn1
        if isempty(nextLine) || ~ischar(nextLine); break;
        else
            LabelOpxName{next,1} = nextLine;
            next = next + 1;
            nextLine = fgetl(fid);
        end
    end
    FileNames.LabelOpxName = LabelOpxName;
case '# Olivine Analysis Files'
    OlAnlName = {};
    GoOn1 = 1; next = 1;
    nextLine = fgetl(fid);
    while GoOn1
        if isempty(nextLine) || ~ischar(nextLine); break;
        else
            OlAnlName{next,1} = nextLine;
            next = next + 1;
            nextLine = fgetl(fid);
        end
    end
    FileNames.OlAnlName = OlAnlName;
case '# Opx Analysis Files'
    OpxAnlName = {};
    GoOn1 = 1; next = 1;
    nextLine = fgetl(fid);
    while GoOn1
        if isempty(nextLine) || ~ischar(nextLine); break;
        else
            OpxAnlName{next,1} = nextLine;
            next = next + 1;
            nextLine = fgetl(fid);
        end
    end
    FileNames.OpxAnlName = OpxAnlName;
end
if ~ischar(nextLine) && nextLine == -1;
    GoOn0 = 0;
end
end

end

% ----- %
function FinalImage = Tif3DReader(Dir, FileTif, varargin)

```

```

if ~isempty(varargin)
    if strcmp(varargin, 'Flip');
        flipSwitch = 1;
    else
        flipSwitch = 0;
    end
    if strcmp(varargin{1}, 'Plot')
%       cmd = varargin{1};
        islice = varargin{2};
        if ischar(islice) && strcmp(varargin{2}, 'All')
            else
                islice = varargin{2};
            end
        end
    end
else
    flipSwitch = 0;
end

% FileTif='rec_scoba_12_200x200x200_sample8_pc-melt_final.tif';
InfoImage=imfinfo([Dir, FileTif]);
mImage=InfoImage(1).Width;
nImage=InfoImage(1).Height;
NumberImages=length(InfoImage);
FinalImage=zeros(nImage,mImage,NumberImages,'uint16');

TifLink = Tiff([Dir, FileTif], 'r');
for i=1:NumberImages
    TifLink.setDirectory(i);
    FinalImage(:,:,i)=TifLink.read();
end
TifLink.close();

if flipSwitch
    for iz = 1 : size(FinalImage, 3)
        FinalImage(:,:,iz) = FinalImage(:,:,iz)';
    end
end
% FinalImage = double(FinalImage);

% getting the dimensions of the sample
% xloc = strfind(FileTif, 'x');
% xDim = str2num(FileTif(xloc(1)-3:xloc(1)-1));
% yDim = str2num(FileTif(xloc(2)-3:xloc(2)-1));
% zDim = str2num(FileTif(xloc(2)+1:xloc(2)+3));

% % Imported this section from online code
% % http://people.ece.cornell.edu/land/PROJECTS/Reconstruction/index.html
% %patch smoothing factor
% rfactor = 0.125;
% %isosurface size adjustment
% level = .8;
% %useful string constants
% c2 = 'facecolor';
% c1 = 'edgecolor';
%
% p=patch(isosurface(smooth3(FinalImage==1), level));
% reducepatch(p,rfactor)
% set(p,c2,[1,0,0],c1,'none');
%
% p=patch(isosurface(smooth3(FinalImage==2), level));
% reducepatch(p,rfactor)
% set(p,c2,[0,1,0],c1,'none');
% % spy(FinalImage(:,:,islice));
% [Xi, Yi, Zi] = meshgrid(0:1:xDim-1, 0:1:yDim-1, 0:1:zDim-1);
%
% % Xi = uint8(Xi);
% % Yi = uint8(Yi);
% % Zi = uint8(Zi);
% % fidbl = double(FinalImage);
% % figure(1); clf;
% % ImageData2D = FinalImage(:,:,islice);
% % fv = isosurface(fidbl, Xi, Yi, Zi);

```



```

% % slice(FinalImage, Xi, Yi, Zi);
% % colormap(jet);
% % bwi = im2bw(FinalImage(:,:,islice));
% % image(bwi);
% if ~isempty(varargin)
%     image(FinalImage(:,:,islice));
% end
% colormap(jet);

end

% ----- %

Download ellipsoid_fit.m from http://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit/content/ellipsoid\_fit.m

% ----- %

function [inpts, outpts, tf] = inoutEllipGen2(x, pars)

P = [pars(1), pars(4), pars(5); ...
     pars(4), pars(2), pars(6); ...
     pars(5), pars(6), pars(3)];

% P = [pars(2), pars(4), pars(5); ...
%      pars(4), pars(1), pars(6); ...
%      pars(5), pars(6), pars(3)];

M1 = P*x';
M2 = sum((x').*M1, 1);

In = M2 <= 1;
Out = M2 > 1;

tf = In;

inpts = x(In,:);
outpts = x(Out,:);

end

% ----- %

```

References

- Aharonov, E., Whitehead, J.A., Kelemen, P.B., Spiegelman, M., 1995. Channeling instability of upwelling melt in the mantle. *J. Geophys. Res.* 100, 20433–20450. doi:10.1029/95JB01307
- Allègre, C.J., Condomines, M., Allègre, C.J., 1982. Basalt genesis and mantle structure studied through Th-isotope geochemistry. *Nature* 299.
- Allègre, C.J., Montigny, R., Bottinga, Y., 1973. Cortège ophiolitique et cortège océanique, géochimie comparée et mode de genèse. *Bull. Soc. Geol. Fr* 15, 7.
- Allwright, J., Katz, R., 2014. Pipe Poiseuille flow of viscously anisotropic, partially molten rock. *arXiv Prepr. arXiv1404.6366* 1–21.
- Andrä, H., Combaret, N., Dvorkin, J., Glatt, E., Han, J., Kabel, M., Keehm, Y., Krzikalla, F., Lee, M., Madonna, C., Marsh, M., Mukerji, T., Saenger, E.H., Sain, R., Saxena, N., Ricker, S., Wiegmann, A., Zhan, X., 2013. Digital rock physics benchmarks—part II: Computing effective properties. *Comput. Geosci.* 50, 33–43. doi:10.1016/j.cageo.2012.09.008
- Asimow, P., Hirschmann, M., Ghiorso, M., O'Hara, M., Stolper, E., 1995. The effect of pressure-induced solid-solid phase transitions on decompression melting of the mantle. *Geochim. Cosmochim. Acta* 59, 4489–4506.
- Avellaneda, M., Torquato, S., 1991. Rigorous link between fluid permeability, electrical conductivity, and relaxation times for transport in porous media. *Phys. Fluids A Fluid Dyn.* 3, 2529. doi:10.1063/1.858194
- Beucher, S., 1992. The watershed transformation applied to image segmentation. *Scanning Microsc.* 6, 299–314.
- Beucher, S., Meyer, F., 1992. The morphological approach to segmentation: the watershed transformation, in: *Optical Engineering*. Marcel Dekker Incorporated, New York, pp. 433–481.
- Bourdon, B., Sims, K., 2003. U-series Constraints on Intraplate Basaltic Magmatism. *Rev. Mineral. Geochemistry* 52, 215–254.
- Boyd, F.R., England, J.L., 1960. Apparatus for phase-equilibrium measurements at pressures up to 50 kilobars and temperatures up to 1750°C. *J. Geophys. Res.* 65, 741–748.
- Buades, A., Coll, B., Morel, J., 2005. A non-local algorithm for image denoising. *Comput. Vis. Pattern Recognit.* 2, 60–65.
- Buck, W.R., Su, W., 1989. Focused mantle upwelling below mid-ocean ridges due to feedback between viscosity and melting. *Geophys. Res. Lett.* 16, 641–644.
- Bulau, J.R., Waff, H.S., Tyburczy, J.A., 1979. Mechanical and Thermodynamic Constraints on Fluid Distribution in Partial Melts. *J. Geophys. Res.* 84, 6102–6108. doi:10.1029/JB084iB11p06102
- Cheadle, M.J., 1989. Properties of texturally equilibrated two-phase aggregates. University of Cambridge.
- Cheadle, M.J., Elliott, M.T., McKenzie, D., 2004. Percolation threshold and permeability of crystallizing igneous rocks: The importance of textural equilibrium. *Geology* 32, 757. doi:10.1130/G20495.1
- Chorin, A.J., 1967. A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys.* 2, 12–26.

- Cmíral, M., Fitz, J.D., Faul, U.H., Green, D.H., 1998. A close look at dihedral angles and melt geometry in olivine-basalt aggregates: a TEM study. *Contrib. to Mineral. Petrol.* 130, 336–345.
- Condomines, M., Morand, P., Allgre, C.J., 1981. ^{230}Th - ^{238}U radioactive disequilibria in tholeiites from the FAMOUS zone (Mid-Atlantic Ridge, $36^{\circ} 50' \text{N}$): Th and Sr isotopic geochemistry. *Earth Planet. Sci. Lett.* 55, 247–256.
- Connolly, J.A.D., Schmidt, M.W., Solferino, G., Bagdassarov, N., 2009. Permeability of asthenospheric mantle and melt extraction rates at mid-ocean ridges. *Nature* 462, 209–12. doi:10.1038/nature08517
- Constable, S., 2006. SEO3: A new model of olivine electrical conductivity. *Geophys. J. Int.* 166, 435–437. doi:10.1111/j.1365-246X.2006.03041.x
- Cooper, R.F., Kohlstedt, D.L., 1982. Interfacial energies in the olivine-basalt system, in: *High Pressure Research in Geophysics*. Center for Academic Publications, pp. 217–228.
- Cooper, R.F., Kohlstedt, D.L., 1984. Sintering of olivine and olivine-basalt aggregates. *Phys. Chem. Miner.* 11, 5–16. doi:10.1007/BF00309372
- Cooper, R.F., Kohlstedt, D.L., 1984. Solution-precipitation enhanced diffusional creep of partially molten olivine-basalt aggregates during hot-pressing. *Tectonophysics* 107, 207–233.
- Dai, L., Karato, S., 2014a. The effect of pressure on the electrical conductivity of olivine under the hydrogen-rich conditions. *Phys. Earth Planet. Inter.* 232, 51–56.
- Dai, L., Karato, S., 2014b. High and highly anisotropic electrical conductivity of the asthenosphere due to hydrogen diffusion in olivine. *Earth Planet. Sci. Lett.* 408, 79–86.
- Dai, L., Li, H., Li, C., Hu, H., Shan, S., 2010. The electrical conductivity of dry polycrystalline olivine compacts at high temperatures and pressures. *Mineral. Mag.* 74, 849–857.
- Daines, M.J., Kohlstedt, D.L., 1994. The transition from porous to channelized flow due to melt/rock reaction during melt migration. *Geophys. Res. Lett.* 21, 145–148. doi:10.1029/93GL03052
- Daines, M.J., Kohlstedt, D.L., 1997. Influence of deformation on melt topology in peridotites. *J. Geophys. Res.* 102, 10210–10257.
- David, C., 1993. Geometry of flow paths for fluid transport in rocks. *J. Geophys. Res.* 98, 12267. doi:10.1029/93JB00522
- Debye, P., Hückel, E., 1923. De la théorie des électrolytes. i. abaissement du point de congélation et phénomènes associés. *Phys. Zeitschrift* 24, 185–206.
- Dick, H., 1989. Abyssal peridotites, very slow spreading ridges and ocean ridge magmatism. *Geol. Soc. London, Spec. Publ.* 71–105.
- Dick, H.J.B., 1977. Evidence of partial melting in the Josephine peridotite. *Magma Genes.* 59–62.
- Dowd, B.A., Campbell, G.H., Siddons, D.P., Marr, R.B., Nagarkar, V. V., Tipnis, S. V., Axe, L., 1999. Developments in synchrotron x-ray computed microtomography at the National Synchrotron Light Source. *Proc. SPIE* 224–236.
- Duda, A., Koza, Z., Matyka, M., 2011. Hydraulic tortuosity in arbitrary porous media flow. *Phys. Rev. E* 84.
- Evans, R., Tarits, P., Chave, A., White, A., Heinson, G., Filloux, J., Toh, H., Seama,

- N., Utada, H., Booker, J., Unsworth, M., 1999. Asymmetric Electrical Structure in the Mantle Beneath the East Pacific Rise at 17°S. *Science* 286, 752–756.
- Faul, U., Fitz Gerald, J., 1999. Grain misorientations in partially molten olivine aggregates: an electron backscatter diffraction study. *Phys. Chem. Miner.* 26, 187–197. doi:10.1007/s002690050176
- Faul, U.H., 1997. Permeability of partially molten upper mantle rocks from experiments and percolation theory. *J. Geophys. Res.* 102, 10299–10311. doi:10.1029/96JB03460
- Faul, U.H., 2000. Constraints on the melt distribution in anisotropic polycrystalline aggregates undergoing grain growth, in: *Physics and Chemistry of Partially Molten Rocks*. Springer, pp. 67–92.
- Faul, U.H., Toomey, D.R., Waff, H.S., 1994. Intergranular basaltic melt is distributed in thin, elongated inclusions. *Geophys. Res. Lett.* 21, 29–32. doi:10.1029/93GL03051
- Fitzgerald, R., 2000. Phase-Sensitive X-Ray Imaging. *Phys. Today* 53, 23–26. doi:10.1063/1.1292471
- Frank, F., 1968. Two-component flow model for convection in the Earth's upper mantle. *Nature* 220, 350–352.
- Frey, F.A., Green, D.H., 1974. The mineralogy, geochemistry and origin of Iherzolite inclusions in Victorian basanites. *Geochim. Cosmochim. Acta*. doi:10.1016/0016-7037(74)90003-9
- Fusseis, F., Schrank, C., Liu, J., Karrech, A., Llana-Fúnez, S., Xiao, X., Regenauer-Lieb, K., 2012. Pore formation during dehydration of a polycrystalline gypsum sample observed and quantified in a time-series synchrotron X-ray microtomography experiment. *Solid Earth* 3, 71–86. doi:10.5194/se-3-71-2012
- Fusseis, F., Xiao, X., Schrank, C., De Carlo, F., 2014. A brief guide to synchrotron radiation-based microtomography in (structural) geology and rock mechanics. *J. Struct. Geol.* 65, 1–16. doi:10.1016/j.jsg.2014.02.005
- Gaetani, G.A., O'Leary, J.A., Koga, K.T., Hauri, E.H., Rose-Koga, E.F., Monteleone, B.D., 2014. Hydration of mantle olivine under variable water and oxygen fugacity conditions. *Contrib. to Mineral. Petrol.* 167, 1–14. doi:10.1007/s00410-014-0965-y
- Garapić, G., Faul, U.H., Brisson, E., 2013. High-resolution imaging of the melt distribution in partially molten upper mantle rocks: evidence for wetted two-grain boundaries. *Geochemistry, Geophys. Geosystems* 14, 1–11. doi:10.1029/2012GC004547
- Garboczi, E.J., 1998. Finite element and finite difference programs for computing the linear electric and elastic properties of digital images of random materials. Building and Fire Research Laboratory, National Institute of Standards and Technology.
- Gardés, E., Gaillard, F., Tarits, P., 2014. Toward a unified hydrous olivine electrical conductivity law. *Geochemistry, Geophys. Geosystems* 4984–5000. doi:10.1002/2014GC005496. Received
- Gray, W.G., 1975. A derivation of the equations for multi-phase transport. *Chem. Eng. Sci.* 30, 229–233.
- Gurmani, S.F., Jahn, S., Brasse, H., Schilling, F.R., 2011. Atomic scale view on

- partially molten rocks: Molecular dynamics simulations of melt-wetted olivine grain boundaries. *J. Geophys. Res. Solid Earth* 116, 1–9. doi:10.1029/2011JB008519
- Gürsoy, D., De Carlo, F., Xiao, X., Jacobsen, C., 2014. TomoPy: a framework for the analysis of synchrotron tomographic data. *Synchrotron Radiat.* 21.
- Hammond, W.C., Humphreys, E.D., 2000. Upper mantle seismic wave velocity: Effects of realistic partial melt geometries. *J. Geophys. Res.* 105, 10975. doi:10.1029/2000JB900041
- Harlow, F.H., Welch, J.E., 1965. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Phys. Fluids* 8, 2182–2189. doi:10.1063/1.1761178
- Hashin, Z., Shtrikman, S., 1963. A variational approach to the theory of the elastic behaviour of multiphase materials. *J. Mech. Phys. Solids* 11, 127–140. doi:10.1016/0022-5096(63)90060-7
- Hays, J.F., 1966. Lime-alumina-silica. *Carnegie Inst. Washingt. Yearb.* 65, 234–239.
- Hier-Majumder, S., Ricard, Y., Bercovici, D., 2006. Role of grain boundaries in magma migration and storage. *Earth Planet. Sci. Lett.* 248, 735–749. doi:10.1016/j.epsl.2006.06.015
- Holness, M., Graham, C., 1991. Equilibrium dihedral angles in the system H₂O - CO₂- NaCl-calcite, and implications for fluid flow during metamorphism. *Contrib. to Mineral. Petrol.* 108, 368–383.
- Holtzman, B.K., Kohlstedt, D.L., 2007. Stress-driven Melt Segregation and Strain Partitioning in Partially Molten Rocks: Effects of Stress and Strain. *J. Petrol.* 48, 2379–2406. doi:10.1093/petrology/egm065
- Holtzman, B.K., Kohlstedt, D.L., Zimmerman, M.E., Heidelbach, F., Hiraga, T., Hustoft, J., 2003. Melt segregation and strain partitioning: implications for seismic anisotropy and mantle flow. *Science* 301, 1227–30. doi:10.1126/science.1087132
- Hughes, M.P., 2000. AC electrokinetics: applications for nanotechnology. *Nanotechnology* 11, 124.
- Iwamori, H., 1994. 238U-230Th-226Ra and 235U-231pa disequilibria produced by mantle melting with porous and channel flows. *Earth Planet. Sci. Lett.* 125, 1–16.
- Johannes, W., Bell, P., Mao, H., Boettcher, A., Chopman, D., Hays, J., Newton, R., Seifert, F., 1971. An Interlaboratory Comparison of Piston-Cylinder Pressure Calibration Using the Albite-Breakdown Reaction. *Contrib. to Mineral. Petrol.* 32, 24–38.
- Johnson, K., Dick, H.J.B., 1992. Open system melting and temporal and spatial variation of peridotite and basalt at the Atlantis II fracture zone. *J. Geophys. Res. Solid Earth* 97, 9219–9241. doi:10.1029/92JB00701
- Johnson, D.L., Koplik, J., Schwartz, L.M., 1986. New pore-size parameter characterizing transport in porous media. *Phys. Rev. Lett.* 57, 2564–2567. doi:10.1103/PhysRevLett.57.2564
- Jones, A.G., Fulla, J., Evans, R.L., Muller, M.R., 2012. Water in cratonic lithosphere: Calibrating laboratory-determined models of electrical conductivity of mantle minerals using geophysical and petrological observations. *Geochemistry, Geophys. Geosystems* 13, 1–27. doi:10.1029/2012GC004055

- Jurewicz, S., Jurewicz, A., 1986. Distribution of Apparent Angles on Random Sections With Emphasis of Dihedral Angle Measurements. *J. Geophys. Res.* 91, 9277–9282.
- Jurewicz, S., Watson, E., 1984. Distribution of partial melt in a felsic system: the importance of surface energy. *Contrib. to Mineral. Petrol.* 85, 25–29. doi:10.1007/BF00380218
- Jurewicz, S., Watson, E., 1985. The distribution of partial melt in a granitic system: The application of liquid phase sintering theory. *Geochim. Cosmochim. Acta* 49, 1109–1121. doi:10.1016/0016-7037(85)90002-X
- Kak, A.C., Slaney, M., 1988. Principles of computerized tomographic imaging. *Siam.*
- Katz, A.J., Thompson, A.H., 1987. Prediction of rock electrical conductivity from mercury injection measurements. *J. Geophys. Res.* 92, 599. doi:10.1029/JB092iB01p00599
- Katz, R.F., Spiegelman, M., Holtzman, B., 2006. The dynamics of melt and shear localization in partially molten aggregates. *Nature* 442, 676–9. doi:10.1038/nature05039
- Kelemen, P.B., Dick, H.J.B., Quick, J.E., 1992. Formation of harzburgite by pervasive melt/rock reaction in the upper mantle. *Nature* 358, 635–641.
- Kelemen, P.B., Hirth, G., Shimizu, N., Spiegelman, M., Dick, H.J., 1997. A review of melt migration processes in the adiabatically upwelling mantle beneath oceanic spreading ridges. *Philos. Trans. R. Soc. London. Ser. A Math. Phys. Eng. Sci.* 355, 283–318.
- Kelemen, P.B., Shimizu, N., Salters, V.J.M., 1995a. Extraction of mid-ocean-ridge basalt from the upwelling mantle by focused flow of melt in dunite channels. *Nature* 375, 747 – 753. doi:10.1038/375747a0
- Kelemen, P.B., Whitehead, J.A., Aharonov, E., Jordahl, K.A., 1995b. Experiments on flow focusing in soluble porous media, with applications to melt extraction from the mantle. *J. Geophys. Res.* 100, 475–496. doi:10.1029/94JB02544
- Key, K., Constable, S., Liu, L., Pommier, A., 2013. Electrical image of passive mantle upwelling beneath the northern East Pacific Rise. *Nature* 495, 499–502. doi:10.1038/nature11932
- King, D.S.H., Hier-Majumder, S., Kohlstedt, D.L., 2011a. An experimental study of the effects of surface tension in homogenizing perturbations in melt fraction. *Earth Planet. Sci. Lett.* 307, 349–360. doi:10.1016/j.epsl.2011.05.009
- King, D.S.H., Holtzman, B.K., Kohlstedt, D.L., 2011b. An experimental investigation of the interactions between reaction-driven and stress-driven melt segregation: 1. Application to mantle melt extraction. *Geochemistry, Geophys. Geosystems* 12, 1-16. doi:10.1029/2011GC003684
- Laporte, D., Provost, A., 2000. Equilibrium geometry of a fluid phase in a polycrystalline aggregate with anisotropic surface energies: Dry grain boundaries. *J. Geophys. Res.* 105, 25937–25953. doi:10.1029/2000JB900256
- Lundstrom, C.C., Gill, J., Williams, Q., Perfit, M.R., 1995. Mantle Melting and Basalt Extraction by Equilibrium Porous Flow. *Science* 270, 1958–1961.
- Madden, T., 1976. Random networks and mixing laws. *Geophysics* 41, 1104–1125.
- Markov, K.Z., 2000. Elementary micromechanics of heterogeneous media, in: *Heterogeneous Media*. Springer, pp. 1–162.

- Martys, N., Garboczi, E.J., 1992. Length scales relating the fluid permeability and electrical conductivity in random two-dimensional model porous media. *Phys. Rev. B* 46, 6080–6090.
- Matyka, M., Khalili, A., Koza, Z., 2008. Tortuosity-porosity relation in the porous media flow. *Phys. Rev. E* 78, 1–8. doi:10.1103/PhysRevE.78.026306
- McKenzie, D., 1984. The Generation and Compaction of Partially Molten Rock. *J. Petrol.* 25, 713–765.
- McKenzie, D., 1985. ²³⁰Th-²³⁸U disequilibrium and the melting process beneath ridge axes. *Earth Planet. Sci. Lett.* 72, 149–157.
- McKenzie, D., 2000. Constraints on melt generation and transport from U-series activity ratios. *Chem. Geol.* 162, 81–94. doi:10.1016/S0009-2541(99)00126-6
- McKenzie, D., Bickle, M.J., 1988. The Volume and Composition of Melt Generated by Extension of the Lithosphere. *J. Petrol.* 29, 625–679.
- Meijerink, J.A., van der Vorst, H.A., 1977. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comput.* 31, 148–148. doi:10.1090/S0025-5718-1977-0438681-4
- Mibe, K., Fujii, T., Yasuda, A., 1998. Connectivity of aqueous fluid in the Earth's upper mantle. *Geophys. Res. Lett.* 25, 1233–1236.
- Miller, K.J., Zhu, W., Montési, L.G.J., Gaetani, G.A., 2014. Experimental quantification of permeability of partially molten mantle rock. *Earth Planet. Sci. Lett.* 388, 273–282. doi:10.1016/j.epsl.2013.12.003
- Morgan, F.D., Williams, E.R., Madden, T.R., 1989. Streaming potential properties of westerly granite with applications. *J. Geophys. Res.* 94, 12449. doi:10.1029/JB094iB09p12449
- Münch, B., Trtik, P., Marone, F., Stampanoni, M., 2009. Stripe and ring artifact removal with combined wavelet-Fourier filtering. *EMPA Act.* 17, 34–35. doi:10.1364/OE.17.008567
- Murase, T., McBirney, A., 1973. Properties of some common igneous rocks and their melts at high temperatures. *Geol. Soc. Am. Bull.* 84, 3563–3592. doi:10.1130/0016-7606(1973)84<3563
- Newman, S., Finkel, R.C., Macdougall, J.D., 1983. ²³⁰Th-²³⁸U disequilibrium systematics in ocean tholeiites from 21°N on the East Pacific Rise. *Earth Planet. Sci. Lett.* 65, 17–33.
- Ni, H., Keppler, H., Behrens, H., 2011. Electrical conductivity of hydrous basaltic melts: implications for partial melting in the upper mantle. *Contrib. to Mineral. Petrol.* 162, 637–650. doi:10.1007/s00410-011-0617-4
- Nover, G., 2005. Electrical properties of crustal and mantle rocks - A review of laboratory measurements and their explanation. *Surv. Geophys.* 26, 593–651. doi:10.1007/s10712-005-1759-6
- Ortoleva, P., Chadam, J., Merino, E., Sen, A., 1987. Geochemical self-organization II: the reactive-infiltration instability. *Am. J. Sci.* 287, 1008–1040.
- Oxburgh, E.R., 1980. Heat flow and magma genesis. *Phys. Magmat. Process.* 161–199.
- Paganin, D., Mayo, S.C., Gureyev, T.E., Miller, P.R., Wilkins, S.W., 2002. Simultaneous phase and amplitude extraction from a single defocused image of a homogeneous object. *J. Microsc.* 206, 33–40. doi:10.1046/j.1365-

2818.2002.01010.x

- Park, H., Yoon, D.N., 1985. Effect of Dihedral Angle on the Morphology of Grains in a Matrix Phase ix. *Metall. Trans. A* 16, 923–928.
- Peate, D., Hawkesworth, C.J., 2005. U series disequilibria: insights into mantle melting and the timescales of magma differentiation. *Rev. Geophys.* 43. doi:10.1029/2004RG000154.1.INTRODUCTION
- Pec, M., Holtzman, B.K., Zimmerman, M., Kohlstedt, D.L., 2015. Reaction infiltration instabilities in experiments on partially molten mantle rocks. *Geology* G36611–1.
- Phipps Morgan, J., 1987. Melt Migration Beneath Mid-Ocean Spreading Centers. *Geophys. Res. Lett.* 14, 1238–1241.
- Poe, B.T., Romano, C., Nestola, F., Smyth, J.R., 2010. Electrical conductivity anisotropy of dry and hydrous olivine at 8GPa. *Phys. Earth Planet. Inter.* 181, 103–111. doi:10.1016/j.pepi.2010.05.003
- Presnall, D.C., Simmons, C.L., Porath, H., 1972. Changes in electrical conductivity of a synthetic basalt during melting. *J. Geophys. Res.* 77, 5665. doi:10.1029/JB077i029p05665
- Qi, C., Kohlstedt, D.L., Katz, R.F., Takei, Y., 2014. An experimental test of the viscous anisotropy hypothesis for partially molten rocks. *arXiv Prepr. arXiv1412.0203* 1–21.
- Quick, J.E., 1982. The origin and significance of large, tabular dunite bodies in the Trinity Peridotite, Northern California. *Contrib. to Mineral. Petrol.* 78, 413–422.
- Rabinowicz, M., Nicolas, A., Vigneresse, J.L., 1984. A rolling mill effect in asthenosphere beneath oceanic spreading centers. *Earth Planet. Sci. Lett.* 67, 97–108.
- Renner, J., Viskupic, K., Hirth, G., Evans, B., 2003. Melt extraction from partially molten peridotites. *Geochemistry, Geophys. Geosystems* 4, 8606. doi:10.1029/2002GC000369
- Ribe, N., 1985. The generation and composition of partial melts in the earth's mantle. *Earth Planet. Sci. Lett.* 73, 361–376.
- Ribe, N., 1988. On the dynamics of generalization. *J. Geophys. Res.* 93, 429–436. doi:10.1037/0033-295X.97.4.576
- Ricard, Y., Bercovici, D., Schubert, G., 2001. A two-phase model for compaction and damage: 2. Applications to compaction, deformation, and the role of interfacial surface tension. *J. Geophys. Res.* 106, 8907–8924.
- Richter, F.M., Davis, A.M., DePaolo, D.J., Watson, E.B., 2003. Isotope fractionation by chemical diffusion between molten basalt and rhyolite. *Geochim. Cosmochim. Acta* 67, 3905–3923. doi:10.1016/S0016-7037(03)00174-1
- Roberts, J.J., Tyburczy, J.A., 1999. Partial-melt electrical conductivity: Influence of melt composition. *J. Geophys. Res.* 104, 7055. doi:10.1029/1998JB900111
- Rudnick, R.L., Ionov, D. a., 2007. Lithium elemental and isotopic disequilibrium in minerals from peridotite xenoliths from far-east Russia: Product of recent melt/fluid-rock reaction. *Earth Planet. Sci. Lett.* 256, 278–293. doi:10.1016/j.epsl.2007.01.035

- Ryan, M.P., Blevins, J.Y.K., 1987. The viscosity of synthetic and natural silicate melts and glasses at high temperatures and 1 bar (105 Pascals) pressure and at higher pressures. US Government Printing Office.
- Saenger, E.H., Bohlen, T., 2004. Finite-difference modeling of viscoelastic and anisotropic wave propagation using the rotated staggered grid. *Geophysics* 69, 583–591. doi:10.1190/1.1707078
- Schmeling, H., 1986. Numerical models on the influence of partial melt on elastic, anelastic and electrical properties of rocks. Part II: electrical conductivity. *Phys. Earth Planet. Inter.* 43, 123–136. doi:10.1016/0031-9201(86)90080-4
- Schock, R.N., Duba, A.G., Shankland, T.J., 1989. Electrical conduction in olivine. *J. Geophys. Res. Solid Earth* 94, 5829–5839.
- Schwartz, L., Martys, N., Bentz, D., 1993. Cross-property relations and permeability estimation in model porous media. *Phys. Rev. E* 48, 4584–4591.
- Scott, D.R., Stevenson, D.J., 1989. A self-consistent model of melting, magma migration and buoyancy-driven circulation beneath mid-ocean ridges. *J. Geophys. Res. Solid Earth* 94, 2973–2988.
- Scott, T., Kohlstedt, D.L., 2006. The effect of large melt fraction on the deformation behavior of peridotite. *Earth Planet. Sci. Lett.* 246, 177–187. doi:10.1016/j.epsl.2006.04.027
- Sifré, D., Gardés, E., Massuyeau, M., Hashim, L., Hier-Majumder, S., Gaillard, F., 2014. Electrical conductivity during incipient melting in the oceanic low-velocity zone. *Nature* 509, 81–85. doi:10.1038/nature13245
- Sims, K.W.W., Goldstein, S.J., Blichert-toft, J., Perfit, M.R., Kelemen, P., Fornari, D.J., Michael, P., Murrell, M.T., Hart, S.R., DePalo, D.J., Layne, G., Ball, L., Jull, M., Bender, J., 2002. Chemical and isotopic constraints on the generation and transport of magma beneath the East Pacific Rise. *Geochim. Cosmochim. Acta* 66, 3481–3504.
- Smith, C., 1964. Some Elementary Principles of polycrystalline microstructure. *Metall. Rev.* 9.
- Smith, C.S., 1948. Grains, Phases, and Interfaces: An Interpretation of Microstructure. *Trans. Am. Inst. Min. Metall. Eng.* 175, 15–51. doi:10.1007/s11661-010-0215-5
- Spiegelman, M., 1993. Flow in deformable porous media. Part 2. Numerical analysis—the relationship between shock waves and solitary waves. *J. Fluid Mech.* 247, 39. doi:10.1017/S0022112093000370
- Spiegelman, M., Elliott, T., 1993. Consequences of melt transport for uranium series disequilibrium in young lavas. *Earth Planet. Sci. Lett.* 118, 1–20.
- Spiegelman, M., Kelemen, P.B., Aharonov, E., 2001. Causes and consequences of flow organization during melt transport: The reaction infiltration instability in compactible media. *J. Geophys. Res.* 106, 2061–2077.
- Spiegelman, M., McKenzie, D., 1987. Simple 2-D models for melt extraction at mid-ocean ridges and island arcs. *Earth Planet. Sci. Lett.* 83, 137–152. doi:10.1016/0012-821X(87)90057-4
- Spiegelman, M., Kelemen, P.B., 2003. Extreme chemical variability as a consequence of channelized melt transport. *Geochemistry, Geophys. Geosystems* 4, 1055–1072. doi:10.1029/2002GC000336

- Stolper, E., Walker, D., Hager, B.H., Hays, J.F., 1981. Melt Segregation from Partially Molten Source Regions: The Importance of Melt Density and Source Region Size. *J. Geophys. Res.* 86, 6261–6271.
- Stracke, A., Bourdon, B., McKenzie, D., 2006. Melt extraction in the Earth's mantle: Constraints from U–Th–Pa–Ra studies in oceanic basalts. *Earth Planet. Sci. Lett.* 244, 97–112. doi:10.1016/j.epsl.2006.01.057
- ten Grotenhuis, S.M., Drury, M.R., Spiers, C.J., Peach, C.J., 2005. Melt distribution in olivine rocks based on electrical conductivity measurements. *J. Geophys. Res. Solid Earth* 110, 1–11. doi:10.1029/2004JB003462
- The MELT Seismic Team, 1998. Imaging the Deep Seismic Structure Beneath a Mid-Ocean Ridge: The MELT Experiment. *Science* 280, 1215–1218. doi:10.1126/science.280.5367.1215
- Tombácz, E., Szekeres, M., 2006. Surface charge heterogeneity of kaolinite in aqueous suspension in comparison with montmorillonite. *Appl. Clay Sci.* 34, 105–124. doi:10.1016/j.clay.2006.05.009
- Toomey, D., Wilcock, W., Solomon, S., Hammond, W., JA, O., 1998. Mantle Seismic Structure Beneath the MELT Region of the East Pacific Rise from P and S Wave Tomography. *Science* 280, 1224–1227. doi:10.1126/science.280.5367.1224
- Toramaru, A., Fujii, N., 1986. Connectivity of Melt Phase in a Partially Molten Peridotite. *J. Geophys. Res.* 91, 9239–9252. doi:10.1029/JB091iB09p09239
- Turcotte, D.L., Schubert, G., 2014. *Geodynamics*. Cambridge University Press.
- Vaughan, P.J., Kohlstedt, D.L., 1982. Distribution of the glass phase in hot-pressed, olivine-basalt aggregates: An electron microscopy study. *Contrib. to Mineral. Petrol.* 81, 253–261. doi:10.1007/BF00371679
- Vincent, L., 1993. Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Trans. Image Process.* 2, 176–201.
- Volpe, A., Hammond, P., 1991. 238U-230Th-226Ra disequilibria in young Mount St. Helens rocks: time constraint for magma formation and crystallization. *Earth Planet. Sci. Lett.* 107, 475–486.
- von Bagen, N., Waff, H.S., 1986. Permeabilities, Interfacial Areas and Curvatures of Partially Molten Systems: Results of Numerical Computations of Equilibrium Microstructures. *J. Geophys. Res.* 91, 9261–9276. doi:10.1029/JB091iB09p09261
- W. Williams, R., B. Gill, J., 1989. Effects of partial melting on the uranium decay series. *Geochim. Cosmochim. Acta* 53, 1607–1619. doi:10.1016/0016-7037(89)90242-1
- Waff, H., Bulau, J., 1982. Experimental Determination of Near-Equilibrium Textures in Partially Molten Silicates at High Pressures. *High Press. Res. Geophys.* 229–236.
- Waff, H., Faul, U., 1992. Effects of Crystalline Anisotropy on Fluid Distribution in Ultramafic Partial Melts. *J. Geophys. Res.* 97, 9003–9014.
- Waff, H.S., 1974. Theoretical considerations of electrical conductivity in a partially molten mantle and implications for geothermometry. *J. Geophys. Res.* 79, 4003. doi:10.1029/JB079i026p04003
- Waff, H.S., Bulau, J.R., 1979. Equilibrium Fluid Distribution in an Ultramafic Partial Melt Under Hydrostatic Stress Conditions. *J. Geophys. Res.* 84, 6109–6114.

- Wan, J., Tokunaga, T.K., 2002. Partitioning of clay colloids at air-water interfaces. *J. Colloid Interface Sci.* 247, 54–61. doi:10.1006/jcis.2001.8132
- Wanamaker, B.J., Duba, A.G., 1993. Electrical conductivity of San Carlos olivine along [100] under oxygen- and pyroxene-buffered conditions and implications for defect equilibria. *J. Geophys. Res. Solid Earth* 98, 489–500.
- Wang, D., Mookherjee, M., Xu, Y., Karato, S., 2006. The effect of water on the electrical conductivity of olivine. *Nature* 443, 977–980. doi:10.1038/nature05256
- Wark, D.A., Watson, E.B., 1998. Grain-scale permeabilities of texturally equilibrated, monomineralic rocks. *Earth Planet. Sci. Lett.* 164, 591–605.
- Wark, D.A., Williams, C.A., Watson, E.B., Price, J.D., 2003. Reassessment of pore shapes in microstructurally equilibrated rocks, with implications for permeability of the upper mantle. *J. Geophys. Res.* 108, 1–16. doi:10.1029/2001JB001575
- Watson, E.B., 1999. Lithologic partitioning of fluids and melts. *Am. Mineral.* 84, 1693–1710.
- Watson, E.B., Brenan, J.M., 1987. Fluids in the lithosphere, 1. Experimentally-determined wetting characteristics of CO₂-H₂O fluids and the implications for fluid transport, host-rock physical properties, and fluid inclusion formation. *Earth Planet. Sci. Lett.* 85, 497–515.
- Watson, H.C., Roberts, J.J., 2011. Connectivity of core forming melts: Experimental constraints from electrical conductivity and X-ray tomography. *Phys. Earth Planet. Inter.* 186, 172–182. doi:10.1016/j.pepi.2011.03.009
- Weatherley, S.M., Katz, R.F., 2012. Melting and channelized magmatic flow in chemically heterogeneous, upwelling mantle. *Geochemistry Geophys. Geosystems* 13, Q0AC18. doi:10.1029/2011GC003989
- Weickert, J., Romeny, B.M.T.H., Viergever, M.A., 1998. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans. Image Process.* 7, 398–410.
- Westneat, M.W., Socha, J.J., Lee, W.-K., 2008. Advances in biological structure, function, and physiology using synchrotron X-ray imaging*. *Annu. Rev. Physiol.* 70, 119–142.
- Whitaker, S., 1998. The method of volume averaging. Springer Science & Business Media.
- Wirth, R., 1996. Thin amorphous films (1–2 nm) at olivine grain boundaries in mantle xenoliths from San Carlos, Arizona. *Contrib. to Mineral. Petrol.* 124, 44–54.
- Xu, Y., Shankland, T.J., Duba, A.G., 2000. Pressure effect on electrical conductivity of mantle olivine. *Phys. Earth Planet. Inter.* 118, 149–161.
- York, D., Evensen, N.M., Martínez, M.L., De Basabe Delgado, J., 2004. Unified equations for the slope, intercept, and standard errors of the best straight line. *Am. J. Phys.* 72, 367–375. doi:10.1119/1.1632486
- Yoshino, T., 2010. Laboratory electrical conductivity measurement of mantle minerals. *Surv. Geophys.* 31, 163–206. doi:10.1007/s10712-009-9084-0
- Yoshino, T., Laumonier, M., McIsaac, E., Katsura, T., 2010. Electrical conductivity of basaltic and carbonatite melt-bearing peridotites at high pressures: Implications for melt distribution and melt fraction in the upper mantle. *Earth Planet. Sci. Lett.* 295, 593–602. doi:10.1016/j.epsl.2010.04.050
- Yoshino, T., Matsuzaki, T., Shatskiy, A., Katsura, T., 2009a. The effect of water on

- the electrical conductivity of olivine aggregates and its implications for the electrical structure of the upper mantle. *Earth Planet. Sci. Lett.* 288, 291–300.
- Yoshino, T., Price, J.D., Wark, D.A., Watson, E.B., 2006. Effect of faceting on pore geometry in texturally equilibrated rocks: Implications for low permeability at low porosity. *Contrib. to Mineral. Petrol.* 152, 169–186. doi:10.1007/s00410-006-0099-y
- Yoshino, T., Yamazaki, D., Mibe, K., 2009b. Well-wetted olivine grain boundaries in partially molten peridotite in the asthenosphere. *Earth Planet. Sci. Lett.* 283, 167–173. doi:10.1016/j.epsl.2009.04.007
- Zhan, X., Schwartz, L., Toksöz, M., 2010. Pore-scale modeling of electrical and fluid transport in Berea sandstone. *Geophysics* 75, F135–F142.
- Zhou, S.-A., Brahme, A., 2008. Development of phase-contrast X-ray imaging techniques and potential medical applications. *Phys. Medica* 24, 129–148.
- Zhu, W., Gaetani, G.A., Fusseis, F., Montési, L.G.J., De Carlo, F., 2011. Microtomography of partially molten rocks: three-dimensional melt distribution in mantle peridotite. *Science* 332, 88–91. doi:10.1126/science.1202221
- Zhu, W., Hirth, G., 2003. A network model for permeability in partially molten rocks. *Earth Planet. Sci. Lett.* 212, 407–416. doi:10.1016/S0012-821X(03)00264-4
- Zimmerman, M.E., Zhang, S., Kohlstedt, D.L., Karato, S., 1999. Melt distribution in mantle rocks deformed in shear. *Geophys. Res. Lett.* 26, 1505. doi:10.1029/1999GL900259